

AD-A284 867



TASK: UU03
CDRL: 05155
31 July 1993

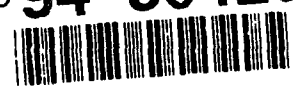
The Reuse-Oriented Software Evolution (ROSE) Process Model Ver 0.5 Draft

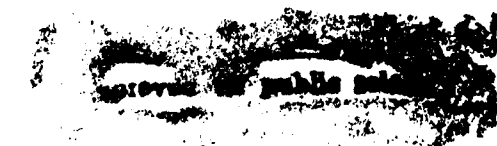
Informal Technical Data

DTIC
ELECTE
SEP 22 1994
S G D

DTIC QUALITY INSPECTED 3

STARS-UC-05155/001/00
31 July 1993

1478 94-30426




94 9 21 000

TASK: UU03
CDRL: 05155
31 July 1993

INFORMAL TECHNICAL REPORT
For
SOFTWARE TECHNOLOGY FOR ADAPTABLE, RELIABLE SYSTEMS
(STARS)

*The Reuse-Oriented Software Evolution (ROSE)
Process Model
Version 0.5 - DRAFT*

STARS-UC-05155/001/0
31 July 1993

Data Type: A005, Informal Technical Data

CONTRACT NO. F19628-93-C-0130
Delivery Order 0011

Prepared for:
Electronic Systems Center
Air Force Systems Command, USAF
Hanscom AFB, MA 01731-5000

Prepared by:
TRW
under contract to
Paramax Systems Corporation
12010 Sunrise Valley Drive
Reston, VA 22091

Accession For	
NTIS	CRA&I <input checked="" type="checkbox"/>
DTIC	TAB <input type="checkbox"/>
Unannounced <input type="checkbox"/>	
Justification	
By	
Distribution /	
Availability Codes	
Dist	Avail and/or Special
A-1	

Distribution Statement "A"
per DoD Directive 5230.24
Authorized for public release; Distribution is unlimited.

DTIC QUALITY INSPECTED 3

Data ID: STARS-UC-05155/001/0

Distribution Statement "A"
per DoD Directive 5230.24
Authorized for public release; Distribution is unlimited.

Copyright 1993, Paramax Systems Corporation, Reston, Virginia
and TRW

Copyright is assigned to the U.S. Government, upon delivery thereto, in accordance with
the DFAR Special Works Clause.

Developed by: TRW under contract to
Paramax Systems Corporation

This document, developed under the Software Technology for Adaptable, Reliable Systems (STARS) program, is approved for release under Distribution "A" of the Scientific and Technical Information Program Classification Scheme (DoD Directive 5230.24) unless otherwise indicated. Sponsored by the U.S. Defense Advanced Research Projects Agency (DARPA) under contract F19628-88-D-0031, the STARS program is supported by the military services, SEI, and MITRE, with the U.S. Air Force as the executive contracting agent.

Permission to use, copy, modify, and comment on this document for purposes stated under Distribution "A" and without fee is hereby granted, provided that this notice appears in each whole or partial copy. This document retains Contractor indemnification to The Government regarding copyrights pursuant to the above referenced STARS contract. The Government disclaims all responsibility against liability, including costs and expenses for violation of proprietary rights, or copyrights arising out of the creation or use of this document.

In addition, the Government, Paramax, and its subcontractors disclaim all warranties with regard to this document, including all implied warranties of merchantability and fitness, and in no event shall the Government, Paramax, or its subcontractor(s) be liable for any special, indirect or consequential damages or any damages whatsoever resulting from the loss of use, data, or profits, whether in action of contract, negligence or other tortious action, arising in connection with the use or performance of this document.

TASK: UU03
CDRL: 05155
31 July 1993

INFORMAL TECHNICAL REPORT
The Reuse-Oriented Software Evolution (ROSE)
Process Model
Version 0.5 - DRAFT

Principal Author(s):

Carol Diane Klingler

Date

Approvals:

Task Manager *Dick Creps*

Date

(Signatures on File)

REPORT DOCUMENTATION PAGE			Form Approved OMB No 0704-0188	
<small>Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204 Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188) Washington, DC 20503.</small>				
1. AGENCY USE ONLY (Leave blank)		2. REPORT DATE 31 July 1993		3. REPORT TYPE AND DATES COVERED Informal Technical Data
4. TITLE AND SUBTITLE The Reuse-Oriented Software Evolution (ROSE) Process Model Version 0.5 - Draft			5. FUNDING NUMBERS F19628-88-D-0031	
6. AUTHOR(S) Carol Diane Klingler, Dick Creps				
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Paramax Systems Corporation 12010 Sunrise Valley Drive Reston, VA 22091			8. PERFORMING ORGANIZATION REPORT NUMBER CDRL Nbr STARS-UC-05155/001/00	
9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES) Department of the Air Force ESC/ENS Hanscom AFB, MA 01731-2816			10. SPONSORING / MONITORING AGENCY REPORT NUMBER 05155	
11. SUPPLEMENTARY NOTES Distribution "C"				
12a. DISTRIBUTION / AVAILABILITY STATEMENT			12b. DISTRIBUTION CODE	
13. ABSTRACT (Maximum 200 words) This document is closely associated with two earlier STARS documents entitled <i>STARS Reuse Concepts Volume I - Conceptual Framework for Reuse Processes (CFRP), Version 2.0</i> [12] and <i>STARS Conceptual Framework for Reuse Processes (CFRP): Application, Version 0.5</i> [13]. These two documents define a conceptual reuse framework and provide example scenarios for use of the framework. This document is intended to follow those documents directly by providing an instantiation and extension of the CFRP. It is thus essential that Reuse Concepts Volume I be read before reading this document. It is strongly recommended that the CFRP: Application document also be read before reading this document.				
14. SUBJECT TERMS			15. NUMBER OF PAGES 129	
			16. PRICE CODE	
17. SECURITY CLASSIFICATION OF REPORT Unclassified	18. SECURITY CLASSIFICATION OF THIS PAGE Unclassified	19. SECURITY CLASSIFICATION OF ABSTRACT Unclassified	20. LIMITATION OF ABSTRACT SAR	

Contents

1	Introduction	2
1.1	Purpose	2
1.2	Scope	2
1.3	Audience	3
1.4	Definitions of Terms	3
1.5	Document Organization	3
2	Context and Motivation	5
3	ROSE Description	8
3.1	Description of Notations	8
3.1.1	Circle Charts	8
3.1.2	Tables	9
3.1.3	IDEF ₀ Diagrams	9
3.1.4	Process Abstraction Hierarchy Diagrams	9
3.2	Overview	10
3.2.1	ROSE Process Categories	13
3.2.2	ROSE IDEF ₀ Context Diagram	18
3.3	Organization Management	19
3.3.1	Plan	23
3.3.1.1	Assess Organization	25
3.3.1.2	Set Direction	26
3.3.1.3	Scope Line of Business	27
3.3.1.4	Plan Infrastructure	32
3.3.1.5	Plan Projects	33
3.3.2	Enact	35
3.3.2.1	Manage Projects	35
3.3.2.2	Implement Infrastructure	38
3.3.3	Learn	41
3.3.3.1	Observe Projects	42
3.3.3.2	Evaluate Projects	44
3.3.3.3	Explore Innovation	44
3.3.3.4	Recommend Enhancements	45
3.4	Domain Engineering	47
3.4.1	Manage Domain Engineering	50
3.4.1.1	Plan Domain Engineering Project	51
3.4.1.2	Enact Domain Engineering Project	52
3.4.1.3	Learn from Domain Engineering Project	54
3.4.2	Perform Domain Engineering	55
3.4.2.1	Analyze and Model Domain	56
3.4.2.2	Develop Software Architecture	63
3.4.2.3	Develop Application Generators	65
3.4.2.4	Develop Software Components	67

3.5	Asset Management	69
3.5.1	Manage Asset Management	72
3.5.1.1	Plan Asset Management Project	73
3.5.1.2	Enact Asset Management Project	73
3.5.1.3	Learn from Asset Management Project	76
3.5.2	Perform Asset Management	77
3.5.2.1	Develop Library Data Model	78
3.5.2.2	Operate Library	80
3.5.2.3	Collect Library Metrics	82
3.5.2.4	Support Assets	83
3.6	Application Engineering	90
3.6.1	Manage Application Engineering	94
3.6.1.1	Plan Application Engineering Project	94
3.6.1.2	Enact Application Engineering Project	97
3.6.1.3	Learn from Application Engineering Project	98
3.6.2	Perform Application Engineering	99
3.6.2.1	Perform Requirements Definition and Analysis	100
3.6.2.1.1	Plan Project	101
3.6.2.1.2	Identify Major Risks	102
3.6.2.1.3	Define Requirements	102
3.6.2.1.4	Analyze Requirements	104
3.6.2.1.5	Engineer System Architecture	106
3.6.2.2	Engineer High-Level Software Architecture	109
3.6.2.2.1	Refine Project Plans	109
3.6.2.2.2	Develop Risk Mitigation Plan	110
3.6.2.2.3	Develop Software Test Plan	110
3.6.2.2.4	Engineer Basic Software Architecture	110
3.6.2.3	Refine Architecture and Engineer Critical Elements	114
3.6.2.3.1	Refine Project Plans	114
3.6.2.3.2	Assess Risks	116
3.6.2.3.3	Engineer Software Architecture	117
3.6.2.3.4	Engineer Critical Element Prototypes	117
3.6.2.3.5	Perform Preliminary Design	118
3.6.2.4	Engineer System	121
3.6.2.4.1	Refine Project Plans	121
3.6.2.4.2	Assess Risks	122
3.6.2.4.3	Perform Detailed Design	122
3.6.2.4.4	Engineer Software Components	125
3.6.2.4.5	Perform Testing and Integration	128
4	Applying the ROSE Process Model	130
4.1	Tailoring	131
4.2	Other Uses of the ROSE PM	133
5	Outstanding Issues	134

List of Figures

1	High Level ROSE PM Structure	10
2	Organization Management Circle Chart	14
3	Domain Engineering Circle Chart	15
4	Asset Management Circle Chart	16
5	Application Engineering Circle Chart	17
6	ROSE IDEF ₀ Context Diagram	18
7	Organization Management Process Abstraction Hierarchy	19
8	Organization Management Table	20
9	Perform Reuse-Based Software Evolution IDEF ₀ Diagram	21
10	Organization Management Planning Process Abstraction Hierarchy	23
11	Plan IDEF ₀ Diagram	24
12	Assess Organization IDEF ₀ Diagram	25
13	Set Direction IDEF ₀ Diagram	27
14	Scope Line of Business IDEF ₀ Diagram	28
15	Plan Product Lines IDEF ₀ Diagram	29
16	Plan Domains IDEF ₀ Diagram	31
17	Select Domain of Focus IDEF ₀ Diagram	32
18	Plan Infrastructure IDEF ₀ Diagram	33
19	Plan Projects IDEF ₀ Diagram	34
20	Enact IDEF ₀ Diagram	36
21	Manage Projects IDEF ₀ Diagram	37
22	Perform Projects IDEF ₀ Diagram	38
23	Implement Infrastructure IDEF ₀ Diagram	39
24	Organization Management Learning Process Abstraction Hierarchy	40
25	Learn IDEF ₀ Diagram	41
26	Observe Projects IDEF ₀ Diagram	42
27	Evaluate Projects IDEF ₀ Diagram	43
28	Explore Innovation IDEF ₀ Diagram	44
29	Recommend Enhancements IDEF ₀ Diagram	46
30	Domain Engineering Process Abstraction Hierarchy	47
31	Domain Engineering Table	48
32	Domain Engineering IDEF ₀ Diagram	49
33	Plan Domain Engineering Project IDEF ₀ Diagram	50
34	Enact Domain Engineering Project IDEF ₀ Diagram	52
35	Manage Domain Engineering Project IDEF ₀ Diagram	53
36	Learn from Domain Engineering Project IDEF ₀ Diagram	54
37	Perform Domain Engineering Project Process Abstraction Hierarchy	55
38	Perform Domain Engineering Project IDEF ₀ Diagram	56
39	Analyze and Model Domain Process Abstraction Hierarchy	57
40	Analyze and Model Domain IDEF ₀ Diagram	58
41	Plan Domain Analysis IDEF ₀ Diagram	59
42	Acquire Domain Knowledge IDEF ₀ Diagram	60
43	Model Domain IDEF ₀ Diagram	61
44	Validate Domain Model IDEF ₀ Diagram	63

45	Develop Software Architecture IDEF ₀ Diagram	64
46	Develop Application Generators IDEF ₀ Diagram	66
47	Develop Software Components IDEF ₀ Diagram	67
48	Asset Management Process Abstraction Hierarchy	69
49	Asset Management Table	70
50	Asset Management IDEF ₀ Diagram	71
51	Plan Asset Management Project IDEF ₀ Diagram	72
52	Enact Asset Management Project IDEF ₀ Diagram	74
53	Manage Asset Management Project IDEF ₀ Diagram	75
54	Learn from Asset Management Project IDEF ₀ Diagram	76
55	Perform Asset Management Project Process Abstraction Hierarchy	77
56	Perform Asset Management Project IDEF ₀ Diagram	78
57	Develop Library Data Model IDEF ₀ Diagram	79
58	Operate Library IDEF ₀ Diagram	81
59	Collect Library Metrics IDEF ₀ Diagram	82
60	Support Assets IDEF ₀ Diagram	83
61	Acquire Assets IDEF ₀ Diagram	84
62	Accept Assets IDEF ₀ Diagram	85
63	Catalog Assets IDEF ₀ Diagram	87
64	Collect Asset Metrics IDEF ₀ Diagram	88
65	Certify Assets IDEF ₀ Diagram	89
66	Application Engineering Process Abstraction Hierarchy	90
67	Application Engineering Table - Part 1	92
68	Application Engineering Table - Part 2	93
69	Application Engineering IDEF ₀ Diagram	94
70	Plan Application Engineering Project IDEF ₀ Diagram	95
71	Enact Application Engineering Project IDEF ₀ Diagram	96
72	Manage Application Engineering Project IDEF ₀ Diagram	97
73	Learn from Application Engineering Project IDEF ₀ Diagram	99
74	Perform Application Engineering Project IDEF ₀ Diagram	100
75	Perform Requirements Definition and Analysis Process Abstraction Hierarchy	101
76	Perform Requirements Definition and Analysis IDEF ₀ Diagram	102
77	Plan Project IDEF ₀ Diagram	103
78	Define Requirements IDEF ₀ Diagram	104
79	Analyze Requirements IDEF ₀ Diagram	105
80	Select and Develop Requirements Model IDEF ₀ Diagram	106
81	Engineer System Architecture IDEF ₀ Diagram	107
82	Select and Develop System Architecture IDEF ₀ Diagram	108
83	Engineer High-Level Software Architecture Process Abstraction Hierarchy	109
84	Engineer High-Level Software Architecture IDEF ₀ Diagram	110
85	Refine Project Plans IDEF ₀ Diagram	111
86	Engineer Basic Software Architecture IDEF ₀ Diagram	112
87	Engineer High-Level Architecture IDEF ₀ Diagram	113
88	Select and Develop High Level Architecture Architecture IDEF ₀ Diagram	114

89	Refine Architecture and Engineer Critical Elements Process Abstraction Hierarchy	115
90	Refine Architecture and Engineer Critical Elements IDEF ₀ Diagram	116
91	Refine Project Plans IDEF ₀ Diagram	117
92	Revise Plans IDEF ₀ Diagram	118
93	Engineer Software Architecture IDEF ₀ Diagram	119
94	Engineer Critical Element Prototypes IDEF ₀ Diagram	120
95	Perform Preliminary Design IDEF ₀ Diagram	121
96	Select and Develop Preliminary Design IDEF ₀ Diagram	122
97	Engineer System Process Abstraction Hierarchy	123
98	Engineer System IDEF ₀ Diagram	124
99	Refine Project Plans IDEF ₀ Diagram	125
100	Perform Detailed Design IDEF ₀ Diagram	126
101	Select and Develop Detailed Design IDEF ₀ Diagram	127
102	Engineer Software Components IDEF ₀ Diagram	128
103	Select and Develop Components IDEF ₀ Diagram	129
104	Perform Testing and Integration IDEF ₀ Diagram	130

Prologue

This document, *The Reuse-Oriented Software Evolution (ROSE) Process Model, Version 0.5*, is currently in draft form. The ROSE model presented in the diagrams within the document is considered complete, but may contain inconsistencies of various kinds. In addition, the textual elaboration and explanation of the model is sketchy or incomplete in places and may also exhibit inconsistencies.

Version 1.0 of this document is planned for release in 1994. It will more fully elaborate on the ROSE model, reflect lessons learned in the application of the model, and reflect input and feedback from reviewers both internal and external to STARS. We thus encourage trial usage of the ROSE model and solicit reader review and feedback as input to Version 1.0 of the document.

The diagrams in this document have been reduced in size to allow them to fit in portrait mode within the text. Full-sized copies of these diagrams may be obtained from the point of contact below.

Please submit comments and requests to:

ROSE Comments and Requests
c/o Dick Creps
Paramax Systems Corporation
Dept. 7670
12010 Sunrise Valley Drive
Reston, VA 22091

Phone: (703) 620-7100
Fax: (703) 620-7916
E-mail: creps@stars.ballston.paramax.com

1 Introduction

This document was developed by a Paramax STARS working group consisting of members from TRW and Paramax, with inputs from the Army STARS Demonstration Project at the Army Communications and Electronics Command (CECOM) in Ft. Monmouth, New Jersey.

This document is closely associated with two earlier STARS documents entitled *STARS Reuse Concepts Volume I - Conceptual Framework for Reuse Processes (CFRP), Version 2.0* [12] and *STARS Conceptual Framework for Reuse Processes (CFRP): Application, Version 0.5* [13]. These two documents define a conceptual reuse process framework and provide example scenarios illustrating how the framework can be used. This document complements those documents directly by specializing and extending the CFRP to form a CFRP-based life-cycle process model. It is thus essential that Reuse Concepts Volume I be read before reading this document. It is strongly recommended that the CFRP: Application document also be read before reading this document.

Throughout the remainder of this document, Reuse Concepts Volume I will be referred to informally as the "CFRP Definition" document, and CFRP: Application will be referred to as the "CFRP Application" document.

1.1 Purpose

The purpose of this document is to bridge the gap between the high-level, general CFRP and detailed, prescriptive process methods. The Reuse-Oriented Software Evolution Process Model (ROSE PM) both specializes and extends the CFRP by adding non-reuse related software engineering processes. As an specialization of the CFRP, the ROSE PM described in this document can be:

- followed as a prescriptive model to perform reuse-oriented software development and evolution;
- tailored to add a particular organization's detailed processes to carry out the activities in the ROSE PM.
- tailored to adapt to a particular organizational structure or set of constraints;
- used as an example to guide other applications of the CFRP.

1.2 Scope

The concepts described in this document are generic with respect to organizations, software engineering approaches, and software engineering technologies and environments.

The scope of this document is limited to providing a high-level process model for software development and evolution. This document does not provide a detailed set of processes for software development and evolution. An organization can integrate the ROSE PM with their already-existing detailed software processes.

1.3 Audience

This document is targeted towards readers having one or more of the following roles in their organizations:

- **Program Planner** – Responsible for planning the objectives, strategy, processes, infrastructure, and resources for software engineering programs or projects. Interested in incorporating domain-specific reuse into those programs/projects.
- **Process Engineer** – Responsible for defining, instantiating, tailoring, monitoring, administering, and evolving software engineering process models. Interested in overall life cycle process models that integrate reuse.

1.4 Definitions of Terms

This section defines key terms that will be used throughout this report.

asset Any unit of information of current or future value to a software-intensive systems development and/or PDSS enterprise. Assets may be characterized in many ways including as software-related work products, software subsystems, software components, contact lists for experts, architectures, domain analyses, designs, documents, case studies, lessons learned, research results, seminal software engineering concepts and presentations, etc.

line of business An application area that is a strategic business focus of an organization.

product line A family of software applications produced by an organization within a line of business, based on a set of functional domains that apply to the application area.

program The set of activities encompassed by (and including) the management of a line of business organization.

project A complete Domain Engineering, Asset Management, or Application Engineering effort that is enacted by Organization Management processes.

1.5 Document Organization

This document is organized as follows:

- Section 1 (this section) provides introductory material that defines the purpose, scope, and audience of the document, as well as definitions of some key terms used throughout the document.
- Section 2 describes the factors that motivated the development of the ROSE PM.
- Section 3 describes the ROSE PM in detail.
- Section 4 discusses how the ROSE PM can be applied.
- Section 5 discusses open issues about the ROSE PM.
- Section 6 lists all documents that are referenced in this volume.

2 Context and Motivation

A number of factors have influenced the development of the ROSE Process Model. The primary motivation has come from other STARS-related process and reuse work. This section explains these motivating factors and also explains where the ROSE model fits in the context of other STARS activities.

STARS has developed the Conceptual Framework for Reuse Processes (CFRP) [12] as a vehicle for understanding and applying the STARS domain-specific reuse-based software engineering paradigm. The CFRP establishes a framework for considering reuse-related software engineering processes, how they interrelate, and how they can be integrated with each other and with non-reuse-related processes to form reuse-oriented life-cycle process models that are tailorable to organization needs. The CFRP is composed of two intersecting "process idioms" entitled *Reuse Management* and *Reuse Engineering*. The Reuse Management idiom focuses on processes for planning, enacting and learning about reuse within an organization. The Reuse Engineering idiom focuses on the processes involved in creating, managing, and utilizing reusable assets. The CFRP defines how these idioms and their lower-level constructs, "process families" and "process categories", interrelate. It also provides a set of principles for integrating these elements to facilitate reuse planning and process modeling.

However, the CFRP by itself is a very general framework. One motivation for developing the ROSE model is the assumption that a typical organization will find the CFRP too general to serve as the sole basis for developing a tailored domain-specific reuse-based software engineering life cycle model. In addition, the CFRP is limited in scope in that it focuses on reuse-related processes and does not adequately address the non-reuse-related aspects of the software engineering life cycle. For example, the CFRP Asset Utilization process family only addresses those aspects of application engineering that involve previously developed assets. It does not address activities that extend beyond reuse, such as configuration management, or software engineering in the absence of previously developed assets.

As an alternative to deriving a comprehensive life-cycle model in a "top-down" fashion, starting with the CFRP, some organizations may attempt a more "bottom-up" approach by building on the increasing number of detailed prescriptive methods that are emerging to support aspects of domain-specific reuse (e.g., Feature-Oriented Domain Analysis (FODA) [5] and the Prieto-Díaz domain analysis method [7]). However, even though these methods provide substantial prescriptive guidance, they are generally too narrowly focused on a small portion of the life cycle and are perhaps too prescriptive and detailed to form the basis for a consistent, comprehensive, and tailorable life cycle model.

The ROSE Process Model was developed by the Paramax STARS team specifically to bridge the gap between the high-level, general framework and the detailed, prescriptive methods. The ROSE Model is a life-cycle process model derived directly from CFRP principles. ROSE specializes the highly generic CFRP by imposing a particular set of assumptions on it in terms of organizational structure and engineering philosophy. ROSE also extends the CFRP by addressing a full range of non-reuse-based life cycle activities, particularly in the areas of program planning and application engineering. One key objective that influenced decisions

about the level of detail and specialization in the ROSE model was that ROSE should remain quite generic and tailorable in its own right. Thus, even though the assumptions inherent in the ROSE model constrain it significantly relative to the CFRP, ROSE nevertheless remains highly adaptable to a wide range of engineering methods, organizational structures, support technologies, and other constraints. It is envisioned that, as ROSE matures, it will form the basis for an easily configurable and adaptable reuse-based software engineering methodology.

Another key objective of the ROSE model was to integrate software maintenance and reengineering in the context of domain-specific reuse to produce a general reuse-oriented approach to software evolution. This objective is consistent with a growing trend in the maintenance and reengineering community to view those disciplines from a more reuse-based perspective [1, 4]. This objective was also motivated by a desire for the ROSE model to be useful to the Army STARS Demonstration Project being supported by Paramax, since the focus of that project is domain engineering in an application reengineering context. In the ROSE Model, software evolution is viewed as inherent in the Plan-Enact-Learn philosophy that pervades the model, so that the same basic set of activities is carried out during evolution as during development. In this context, maintenance and reengineering are most properly addressed from a domain engineering perspective. Maintenance, even in a single-system context, can be viewed as production of a closely-related series of systems (i.e., different versions and variants), encompassing the same set of domains. Furthermore, such systems are typically developed and maintained in the context of broader product lines or families of applications that are similarly related through a common set of domains. A set of coordinated, focused domain engineering efforts can yield great cost savings and quality improvement in such environments.

ROSE has also been influenced by other STARS and non-STARS work. To address the complete application engineering life cycle, ROSE includes material derived from earlier STARS process model work performed by TRW. In particular, many of the non-reuse-oriented application engineering activities in the ROSE model were derived from the STARS Composite Process Model (SCPM) [10] and the Navy Command and Control (C²) Process Model (NCCPM) [14]. The SCPM is a risk-driven spiral model for high-performance, trusted system development, adapted to follow a reuse paradigm. The NCCPM is a tailoring of the SCPM to the Navy Command and Control (C²) Domain. The NCCPM describes an entire system development life cycle from early concept through contract award, design, development, operations, and maintenance.

Another product that has influenced the ROSE model is the STARS Organization Domain Modeling (ODM) [11] domain analysis process model, which is also strongly tied to the CFRP. The ODM model's relationships to the CFRP provided useful insights into domain engineering in the context of the CFRP. The ODM model also proved useful as a checklist and "sanity check" when developing and refining the ROSE Domain Engineering process submodel. The Army STARS Demonstration Project's application of ODM and their preliminary work with ROSE have also provided useful feedback during ROSE development.

Among the non-STARS influences on the ROSE model, Basili's Experience Factory work

[2, 3] stands out as being most influential. In some ways, the basic Experience Factory concepts can be viewed as comparable to the CFRP. The Experience Factory concepts have been refined and implemented in practical settings at NASA-Goddard and other sites. The Experience Factory emphasis on continual learning and improvement is reflected directly in several detailed learning activities in the ROSE model.

It should be noted that the ROSE model is an excellent example of a process model produced through the kinds of activities that are illustrated in the STARS CFRP Application document [13]. That document provides a variety of information that supplements the basic CFRP definition, including a set of scenarios that illustrate how the CFRP could be applied to produce CFRP-based process models, project plans, and so on. Since the ROSE model is the product of a real CFRP application effort, it complements the CFRP Application document by providing a concrete example.

3 ROSE Description

This section will describe the details of the ROSE PM. The ROSE PM is divided into four submodels and hierarchically organized within each submodel. Within each submodel, the top level Plan-Enact-Learn processes are called "process families". At the next level of hierarchical decomposition, processes are called "process categories". Processes at lower levels of decomposition are called "processes" or "activities". This naming convention for different levels within the model is used solely to make it easier to convey information about the different levels of processes within the model. This convention is also used in the CFRP Definition and Application Documents.

Within the text, process and product names that refer to names in a diagram will appear with the first letter of each word capitalized. The first time these process or product names are used to reference a particular diagram, the name will appear in **bold** type to tell the reader that the name can be found in the diagram. Process names (except submodel names) will appear as verbs and product names will appear as nouns.

The remainder of this section will describe the ROSE PM. Section 3.1 will describe the notations used in the diagrams. Section 3.2 will describe the ROSE PM at a high level and explain the relationship between the submodels. Section 3.3 will describe the Organization Management submodel of the ROSE PM. Section 3.4 will describe the Domain Engineering submodel. Section 3.5 will describe the Asset Management submodel. Section 3.6 will describe the Application Engineering submodel.

3.1 Description of Notations

This section describes the notations used in the ROSE PM diagrams. The ROSE PM is presented in four different notations that show different views of the same model. These notations are Circle Charts, Process Tables, IDEF₀ Diagrams, and Process Abstraction Hierarchy Diagrams. The Process Tables are the most compact notation, yet show the most complete decomposition of the ROSE PM activities. The other notations extend to varying levels of decomposition and provide alternative perspectives on the activity structure. The IDEF₀ diagrams also provide substantial additional information concerning information flows among the activities.

3.1.1 Circle Charts

The Circle Charts are intended to provide a high-level overview of each of the ROSE PM submodels. Each Circle Chart shows a graphical, CFRP-derived view of one submodel. The chart is broken into three quadrants, Plan, Enact, and Learn (derived from the Reuse Management idiom of the CFRP), with process category names in each quadrant. Arrows around the outside of the circle depict the general direction of information flow and influence. They also, to some degree, indicate sequencing among the process categories, although sequencing can vary substantially depending on specific circumstances, and activities within different

families and categories often will operate in parallel.

3.1.2 Tables

Each ROSE PM Process Table presents the detailed activities that occur within a ROSE submodel, in a tabular, hierarchical format. In all the submodel tables, the activities are organized into Plan-Enact-Learn process families down the left hand side, with bold horizontal lines separating each family. In the Organization Management submodel table, the left hand side is also organized into process categories within each of the families, with finer horizontal lines separating each category. The individual activities within each category are then listed in rows across the page, and even though they are arranged in columns, no additional specific structure is implied for the activities.

The Domain Engineering, Asset Management, and Application Engineering project submodel tables are organized somewhat differently, reflecting a layering of Plan-Enact-Learn activities. The activities are still organized broadly into top-level Plan-Enact-Learn families along the left hand side, but the categories within the Plan and Learn families (which define planning and learning activities for the project as a whole) are listed in a single row across the page. The Enact family is divided into a top row of activities that pertain to the management and infrastructure implementation for the overall project, and a set of rows defining the categories of project activities that are being enacted. These latter rows are further divided into columns that identify the specific planning, enacting, and learning activities associated with each process category. This structure reflects the CFRP view that at any level of work, including the process category level, explicit activities should be identified to plan and to learn about the work involved.

3.1.3 IDEF₀ Diagrams

An IDEF₀ activity diagram contains one level of decomposition of a process. The decomposition structure mirrors the structure embodied in the process tables, to nearly the same level of depth. Boxes within an IDEF₀ diagram show the subprocesses of the parent process named by the diagram. Arrows between the boxes show the flow of information among processes. Arrows entering the left side of a box are inputs to an activity; arrows exiting the right side of a box are outputs, arrows entering the top of a box are controls, and arrows entering the bottom of a box are mechanisms. A sequential ordering of boxes in a diagram implies some information flow dependencies between the activities, but does not imply a sequential flow of control between the activities. More information about IDEF₀ can be obtained from [6, 8].

3.1.4 Process Abstraction Hierarchy Diagrams

The Process Abstraction Hierarchy Diagrams Notation is used to describe graphically the decomposition of processes. The decomposition structure within these diagrams closely mir-

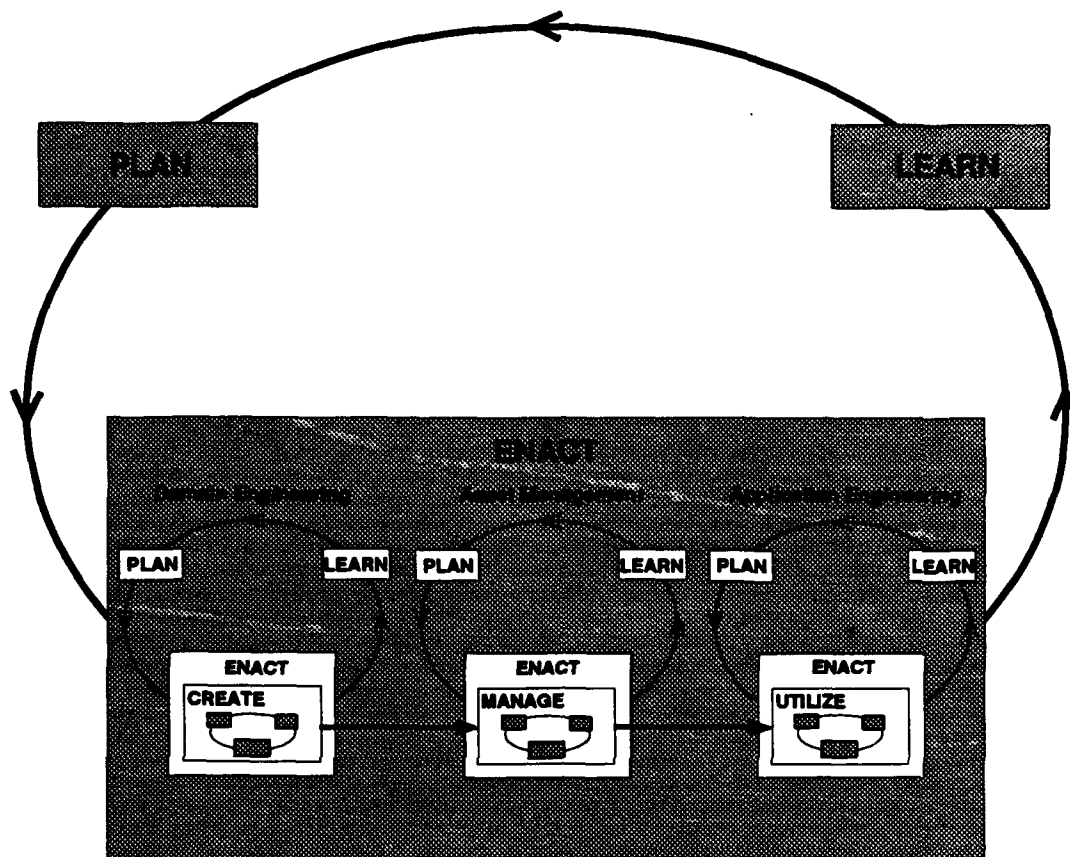


Figure 1: High Level ROSE PM Structure

rors the structure in the process tables, but its graphical presentation provides a different perspective. The process abstraction hierarchy diagram structure directly correlates with that of the IDEF₀ diagrams, and thus nicely complements those diagrams by showing the hierarchical decomposition of processes over a number of IDEF₀ diagrams. Process Abstraction Hierarchy Diagrams appear as trees of named nodes, with each node representing a process at some level of abstraction.

3.2 Overview

As described in Section 2, the ROSE Process Model (ROSE PM) is a specialization and extension of the CFRP. As such, the ROSE PM contains process submodels that interpret the CFRP process idioms in particular ways and extend the CFRP idioms and families to add activities that are not reuse-related. The ROSE PM consists of an Organization Management submodel and three "project" submodels called Domain Engineering, Asset Management, and Application Engineering. Figure 1 shows the relationship between these submodels.

In the canonic view of the CFRP (with the Create-Manage-Utilize idiom embedded within the Enact portion of the Plan-Enact-Learn idiom), the Plan-Enact-Learn idiom can be interpreted as encompassing management of an overall "reuse program" within an organization. This interpretation is reflected in the Organization Management submodel of the ROSE PM. Organization Management encompasses an organization focusing on a single line of business that involves one or more distinct product lines. The Organization Management submodel describes the overall reuse program management activities within such an organization. The Organization Management submodel is depicted by the outer grey-shaded Plan, Enact, and Learn boxes in the figure.

Nested within the **Enact** portion of Organization Management are the types of projects that are carried out by the organization. The Domain Engineering, Asset Management, and Application Engineering project submodels of the ROSE PM describe these types of projects, as derived from the CFRP Asset Creation, Asset Management, and Asset Utilization process families, respectively. A ROSE-consistent organization can include one or more Domain Engineering, Asset Management, or Application Engineering projects.

A **Domain Engineering** project develops domain models, architectures, components, and application generators for a single domain. The Asset Creation activities described in the CFRP Definition document are performed in the Enact process family of Domain Engineering.

An **Asset Management** project develops and operates an asset library to allow storage and retrieval of assets. The Asset Management activities described in the CFRP Definition document are performed in the Enact process family of Asset Management.

An **Application Engineering** project develops a software system within some product line, and may use assets developed in Domain Engineering and stored in Asset Management. The Asset Utilization activities described in the CFRP Definition document are embedded within the activities performed in the Enact process family in Application Engineering. Application Engineering also includes traditional software engineering activities carried out to develop the software system.

The overall organizational perspective embodied in the ROSE PM structure described above reflects a particular set of organizational assumptions imposed on the more general CFRP. However, as noted below and in section 4, the ROSE PM as it is described in this document is "canonic," much like the canonic CFRP noted above, and is readily tailorable and adaptable to constrained sets of alternative assumptions.

The following paragraphs elaborate on some of the key precepts and principles inherent in the ROSE PM.

Pervasiveness of Plan, Enact, Learn

One of the main principles applied in developing the ROSE PM is that the Plan-Enact-Learn idiom described in the CFRP Definition document can be applied at many different levels within an organization, not just at the Organization Management level. Plan-Enact-Learn activities can be defined at any level within the organization where it is useful to explicitly manage decisions within a distinct scope of planning. In the ROSE PM, distinct planning scopes (and thus, explicit Plan-Enact-Learn activities) occur at the Organization Management level, at the level of each individual project (Domain Engineering, Asset Management, and Application Engineering), and at the level of individual categories of engineering activity (i.e., engineering process categories) within a project. At each of these levels, explicit Plan, Enact, and Learn activities are defined.

At each hierarchical level of the ROSE PM, the Plan-Enact-Learn activities at that level will "inherit" planning decisions and artifacts from the higher level scope of planning, and may supplement them to address issues local to the lower level.

Integration of Maintenance and Evolution

In the ROSE PM, maintenance and evolution of assets and software systems is addressed in a consistent, integrated way. Maintenance and evolution is not included as a phase tacked on to the end of the software development life cycle, but as the production of a series of distinct system versions within a product line, using assets within the same set of domains. Therefore, the same general set of steps are carried out in software evolution as in software development, although there may be shifts in emphasis as legacy materials and experience accumulate. By following these steps, maintenance and evolution also receive the benefit inherent in the Plan-Enact-Learn aspects of the domain-specific reuse-based paradigm, with learning occurring from each system modification. No explicit maintenance phase is needed. The same is true for asset evolution and maintenance and asset library evolution and maintenance.

The ROSE PM encourages a Domain Engineering perspective on all application development, maintenance, evolution, or reengineering. The recommended viewpoint is that all software engineering activity should center around model-based analyses that emphasize adaptable, evolvable software architectures and components/generators. Such activity thus focuses principally on the production of *assets* from which application systems are constructed, rather than directly on production of the systems themselves. Hence, software evolution should focus on evolving the assets, rather than the individual legacy systems. This is true even in the context of engineering a single system that will be maintained over time, because that system is in effect a family of systems encompassing its past, present, and future anticipated versions. Those versions can be used as domain exemplars in a domain analysis.

Note, however, that the ROSE Application Engineering submodel also accommodates more conventional reengineering and new development as needed, because domain engineering is not always justifiable economically or for other reasons.

ROSE PM Application Principles

The ROSE PM can be applied as a prescriptive model to carry out reuse-oriented software development and evolution. However, it is more likely that the ROSE PM will be adapted or tailored by an organization before being applied. This section briefly describes some of the anticipated ways in which an organization could apply the ROSE PM. These application principles are described more fully in section 4.

In mapping the ROSE PM into an existing organization, reconfiguration, tailoring, addition, or deletion of processes within the model may be appropriate. The level of granularity of activities identified in the ROSE PM, while much lower than the CFRP, is still high enough that each activity, in general, can be implemented in a wide variety of ways. That is, the ROSE PM, if interpreted literally, prescribes the overall structure for a large set of activities and the information flow among activities, but does not prescribe the methods used to perform those activities at a detailed level. Thus, an organization applying ROSE may impose its own arbitrarily selected set of methods on the activities within the model. For example, one organization may use some form of conventional structured methods within the Develop Domain Architecture activity, while another may opt for an object-oriented approach.

At a different level of consideration, an organization may reject some of the detailed activities in the ROSE PM because they are inconsistent with organization policy or deemed unnecessary. Similarly, they may reject some aspects of the high-level organizational assumptions embodied in the canonic ROSE PM. However, they may accept the overall activity structure of major portions of the ROSE PM and may embrace fundamental ROSE principles wholeheartedly. Such an organization can use the ROSE PM as a basis for adaptation to their own needs by adopting the overall process structure, adjusting it to match organization policy and structure, and elaborating on the model further by applying ROSE principles and other criteria.

For example, ROSE activities can be performed across a number of levels of management or can be distributed across a number of peer organizations. The ROSE PM is adaptable to very rigid or flexible management styles in terms of the amount of planning information that is inherited by lower-level scopes of planning and the degree of autonomy that the lower-level organization has to override or supplement that information. As a more specific example, the ROSE PM assumes that Application Engineering follows DoD Standard 2167A [15], but it can be tailored to organizations following other standards.

3.2.1 ROSE Process Categories

As explained above, each of the ROSE submodels is divided into Plan, Enact, and Learn process families. These families are in turn divided into process categories. ROSE "Circle Charts" show the decomposition of the process categories within the process families of each the submodels. These Circle Charts are described below.

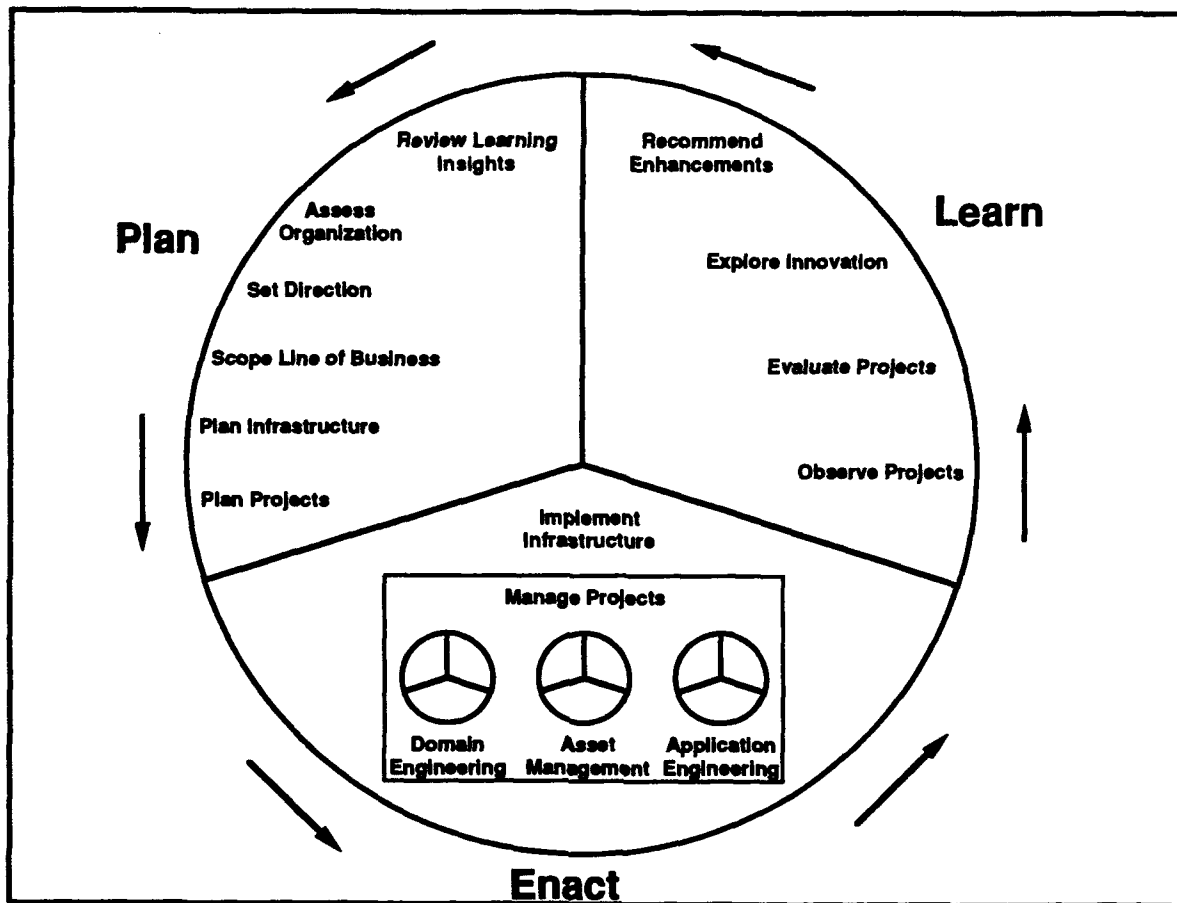


Figure 2: Organization Management Circle Chart

Organization Management The process categories within the Plan, Enact, and Learn process families in Organization Management are shown in the Circle Chart in Figure 2. The Plan process family includes process categories to **Review Learning Insights** from the previous program Plan-Enact-Learn cycle; **Assess** the current state of practice within the **Organization**; **Set** the strategic **Direction** of the organization; **Scope** the boundaries of the organization's **Line of Business**, including its associated product lines and domains; **Plan** the development of the organization's technical, organizational, and educational **Infrastructure**; and **Plan** the **Projects** that are within the organization's scope. The Enact process family includes process categories to **Implement** the **Infrastructure**; and to **Manage** the **Projects** within the organization's scope. The Managed Projects can include Domain Engineering, Asset Management, and Application Engineering projects. The Learn process family includes process categories to **Observe** the processes carried out and products developed in the **Projects** and generate observation data; **Evaluate** the observation data from the **Projects** relative to their planned objectives; **Explore Innovation** to support the evolution of the organization's basic capabilities; and **Recommend Enhancements** to the organization's capabilities as input to the next planning cycle. These activities are described more fully in Section 3.3.

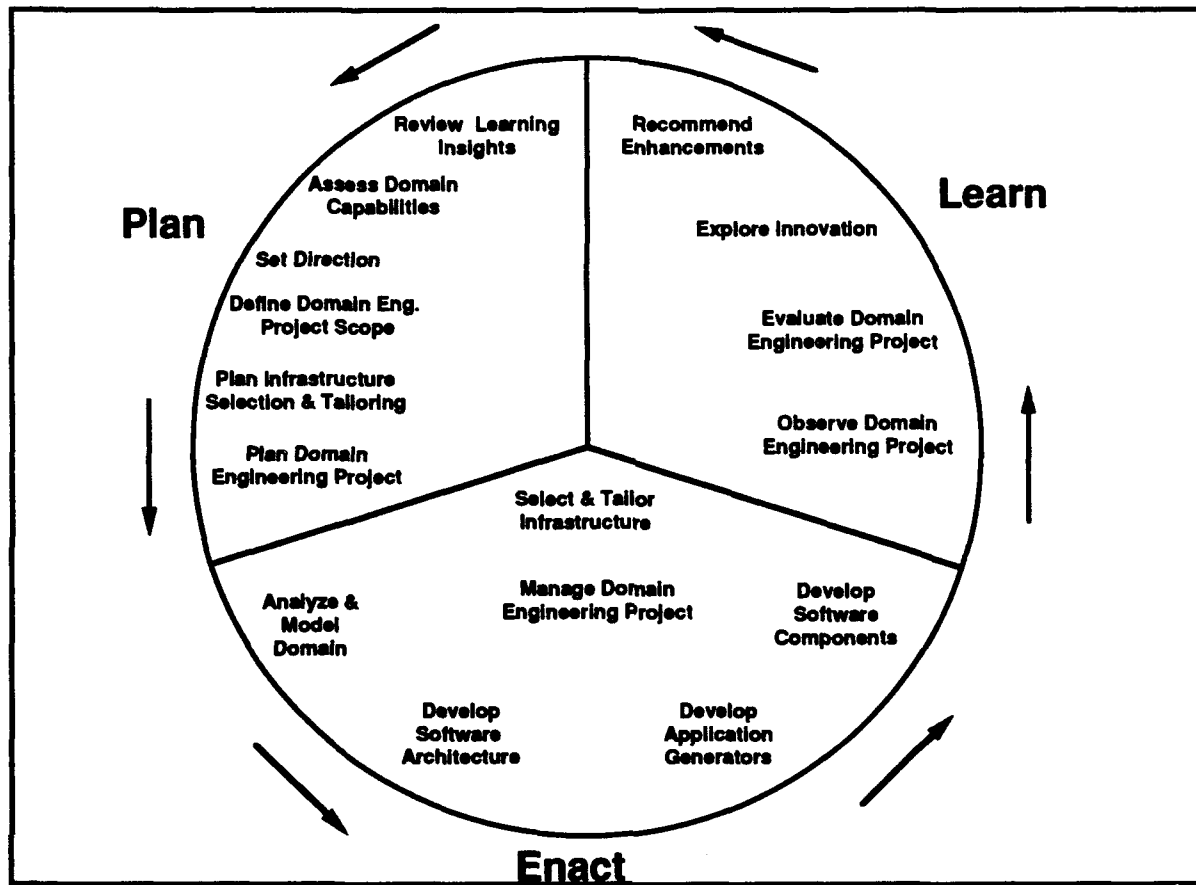


Figure 3: Domain Engineering Circle Chart

Domain Engineering The process categories within the Plan, Enact, and Learn process families in Domain Engineering are shown in the Circle Chart in Figure 3. The Plan process family includes process categories to **Review Learning Insights** from the previous project Plan-Enact-Learn cycle; **Assess** the current **Domain Capabilities** within the organization; **Set** the strategic **Direction** of the Domain Engineering project; **Define Project Scope** for the Domain Engineering project; **Plan** the technical, organizational, and educational **Infrastructure Selection and Tailoring**; and **Plan** budget, schedule, resource needs, etc. of the **Domain Engineering Project**. The Enact process family includes process categories to **Select and Tailor** the **Infrastructure**; **Manage** the performance of the **Project**; **Analyze and Model** the project's **Domain**; **Develop a Software Architecture** for the domain; **Develop Application Generators** that generate software components within the domain; and **Develop Software Components** for the domain. The Learn process family includes process categories to **Observe** the processes carried out and products developed in the **Domain Engineering Project** and generate observation data; **Evaluate** the observation data from the **Domain Engineering Project**; **Explore Innovation** to support the evolution of the project's capabilities; and **Recommend Enhancements** to the project's capabilities. These activities are described more fully in Section 3.4.

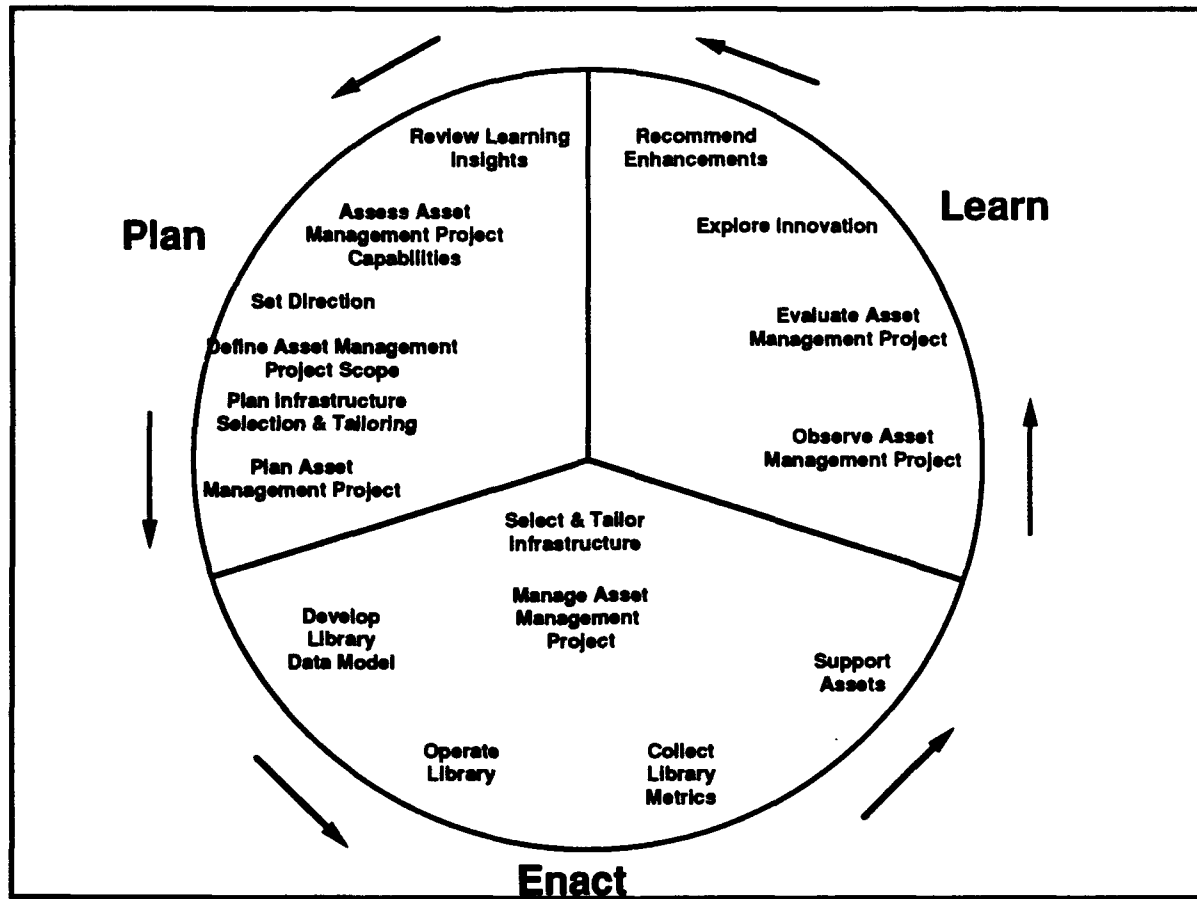


Figure 4: Asset Management Circle Chart

Asset Management The process categories within the Plan, Enact, and Learn process families in Asset Management are shown in the Circle Chart in Figure 4. The Plan process family includes process categories to **Review Learning Insights** from the previous project Plan-Enact-Learn cycle; **Assess** the current **Asset Management Project Capabilities** within the organization; **Set** the strategic **Direction** of the Asset Management project; **Define Project Scope** for the Asset Management project; **Plan** the technical, organizational, and educational **Infrastructure Selection and Tailoring**; and **Plan** budget, schedule, resource needs, etc. of the **Asset Management Project**. The Enact process family includes process categories to **Select and Tailor** the **Infrastructure**; **Manage** the performance of the **Project**; **Develop** the **Library Data Model**; **Operate** the **Library**; **Collect Library Metrics**; and **Support Assets** in the library by selecting assets for inclusion, evaluating assets, cataloging assets, certifying assets, and collecting asset metrics. The Learn process family includes process categories to **Observe** the processes carried out and products developed in the **Asset Management Project** and generate observation data; **Evaluate** the observation data from the **Asset Management Project**; **Explore Innovation** to support the evolution of the project's capabilities; and **Recommend Enhancements** to the project's capabilities. These activities are described more fully in Section 3.5.

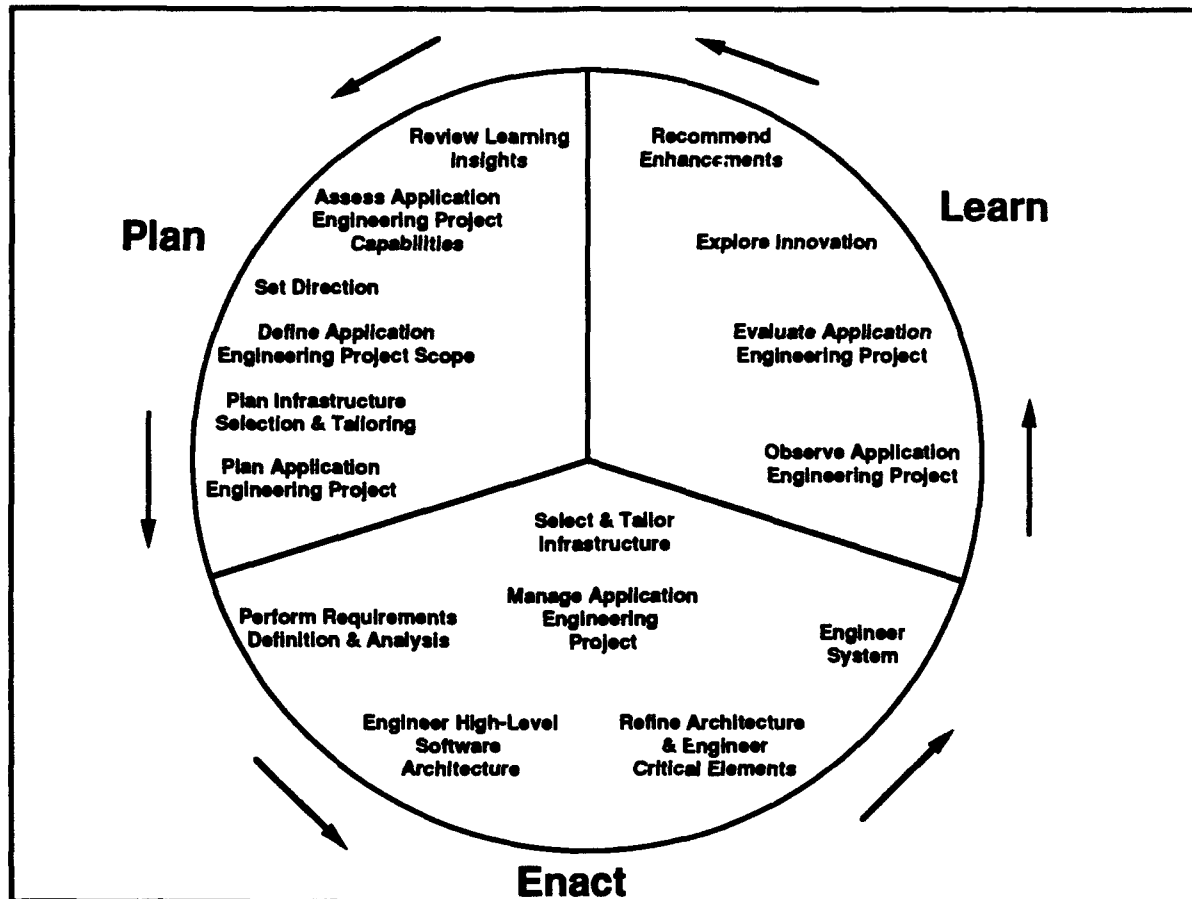
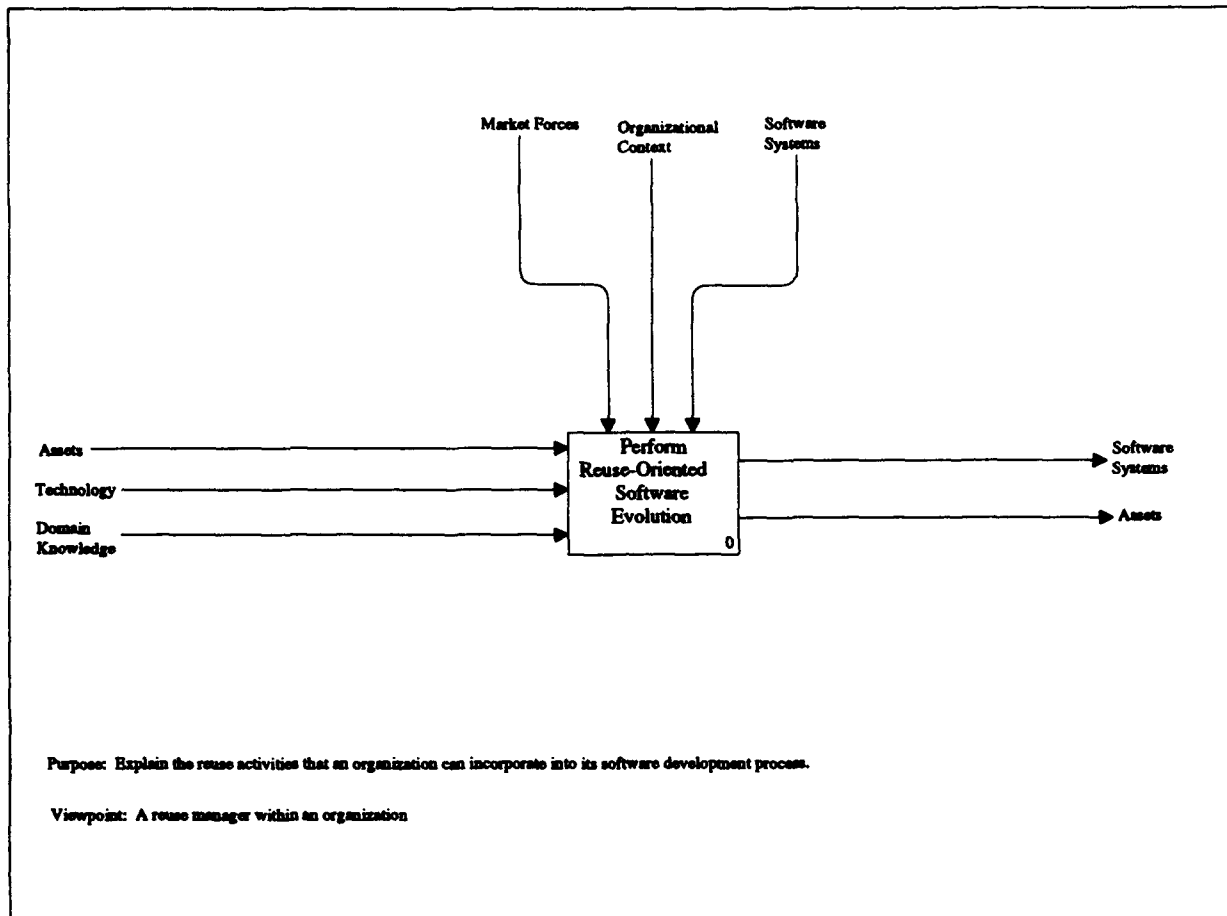


Figure 5: Application Engineering Circle Chart

Application Engineering The process categories within the Plan, Enact, and Learn process families in Application Engineering are shown in the Circle Chart in Figure 5. The Plan process family includes process categories to **Review Learning Insights** from the previous project Plan-Enact-Learn cycle; **Assess** the current **Application Engineering Project Capabilities** within the organization; **Set** the strategic **Direction** of the Application Engineering project; **Define Project Scope** for the Application Engineering project; **Plan** the technical, organizational, and educational **Infrastructure Selection and Tailoring**; and **Plan** budget, schedule, resource needs, etc. of the **Application Engineering Project**. The Enact process family includes process categories to **Select and Tailor the Infrastructure**; **Manage** the performance of the **Project**; **Perform Requirements Definition and Analysis**; **Engineer the High-Level Software Architecture**; **Refine the Architecture and Engineer Critical Components**; and **Engineer the Software System**. The Learn process family includes process categories to **Observe** the processes carried out and products developed in the **Application Engineering Project** and generate observation data; **Evaluate** the observation data from the **Application Engineering Project**; **Explore Innovation** to support the evolution of the project's capabilities; and **Recommend Enhancements** to the project's capabilities. These activities are described more fully in

Figure 6: ROSE IDEF₀ Context Diagram

Section 3.6.

3.2.2 ROSE IDEF₀ Context Diagram

Figure 6 shows the IDEF₀ Context Diagram for the ROSE PM. This Context Diagram shows external inputs, controls, mechanisms, and outputs that interact with the ROSE PM. In the sections that follow, the ROSE PM submodels will be described in detail in terms of additional IDEF₀ diagrams, as well as Process Tables and Process Abstraction Hierarchy Diagrams.

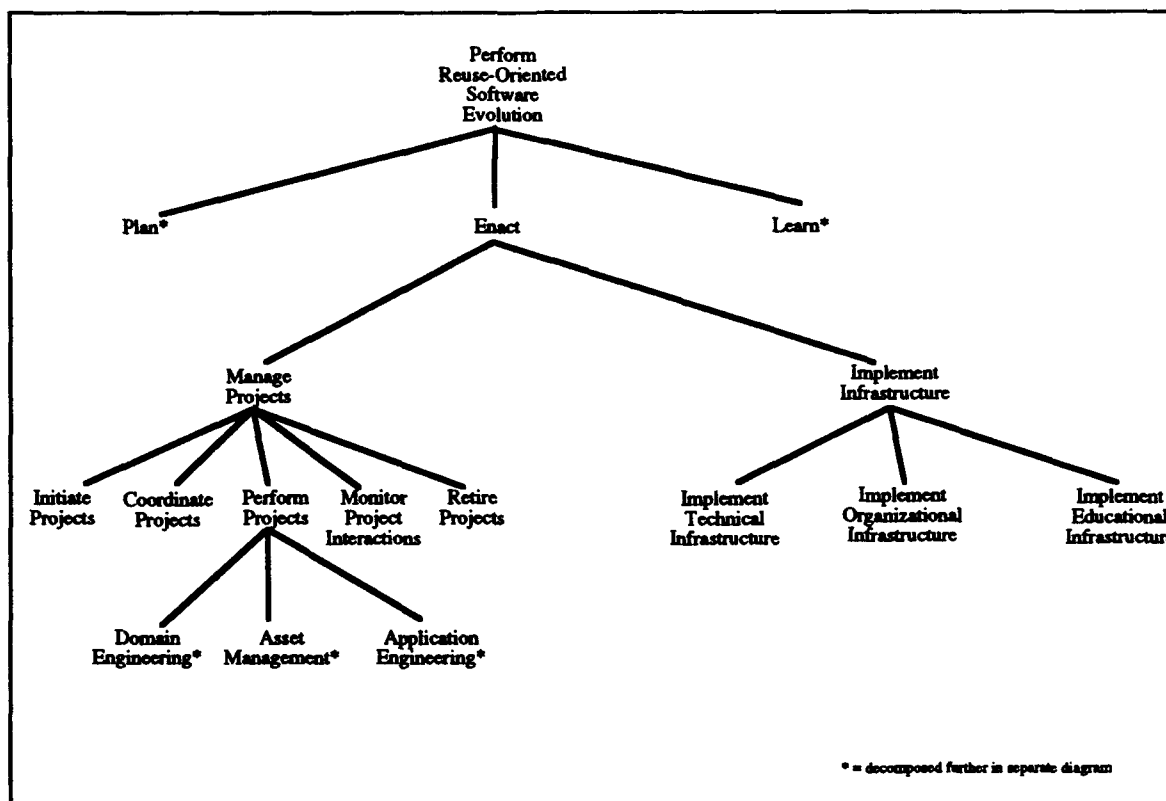


Figure 7: Organization Management Process Abstraction Hierarchy

3.3 Organization Management

Organization Management consists of the highest level organizational Plan, Enact, and Learn activities. In the ROSE model, the project (Domain Engineering, Asset Management, and Application Engineering) activities are hierarchically placed within the Enact portion of Organization Management. This section will discuss the Plan, Enact, and Learn activities within Organization Management. Sections 3.4 through 3.6 will discuss the individual project activities. The process abstraction hierarchy diagram for Organization Management is shown in Figure 7.

Figure 8 shows a table of the activities within Organization Management. The activities are divided into the categories:

- Assess Organization
- Set Direction
- Scope Line of Business

Category	Activities		
Assess Organization	Review Learning Insights	Assess Organizational Practice Assess Process Capabilities Assess Reuse Capabilities Assess Infrastructure Capabilities	Assess Technology Process Analysis Reusability Analysis Tool Analysis
Set Direction	Determine Line of Business Objectives Vision Mission	Determine Capability Objectives Reuse Process Technology	Determine Strategy Determine Criteria for Meeting Objectives
Scope Line of Business	Plan Domains Identify Domains in Line of Business Characterize Domains Select Domains of Focus	Plan Product Lines Identify Product Lines in Line of Business Characterize Product Lines Select Product Lines of Focus	Coordinate Domains & Product Lines
Plan Infrastructure	Plan Technical Infrastructure	Plan Organizational Infrastructure Plan Staffing Availability	Plan Educational Infrastructure
Plan Projects	Select Projects Determine Project Interconnections Set Project Direction Determine Objectives	Plan Project Infrastructure Usage Select Reuse Rewards Perform Detailed Planning	Obtain Commitment Budget Schedule Resource Needs Potential Payoff
Manage Projects	Initiate Projects Coordinate Projects	Perform Projects Domain Engineering Projects Asset Management Projects Application Engineering Projects	Monitor Project Interactions Retire Projects
Implement Infrastructure	Implement Technical Infra. SW Process Descriptions, Methods, Standards Tools Environments	Implement Organizational Infra. Select, tailor, authorize, & communicate policies, procs, & guidelines Determine project staffing	Implement Educational Infra.
Observe Projects	Conduct Observation Reflective Observation Automated Observation Researcher Observations	Solicit Information Gather Metrics Data	Package Experience Generalize Experience Formalize Experience Document Process History
Evaluate Projects	Model Experience Analyze Metrics Data	Compare to Plans & Objectives Learn from Infrastructure Usage	Identify Reusable Processes Review Results
Explore Innovation	Plan Generate Hypotheses Review Observation & Evaluation Data	Enact Test Hypotheses Collect Project Data Perform Experimental Projects Explore External Innovation	Learn Analyze Results Develop Recommendations
Recommend Enhancements	Synthesize Results Analyze Feasibility	Perform Appraisals & Trade Offs Analyze External Needs	Determine Recommendations for Enhancements

Figure 8: Organization Management Table

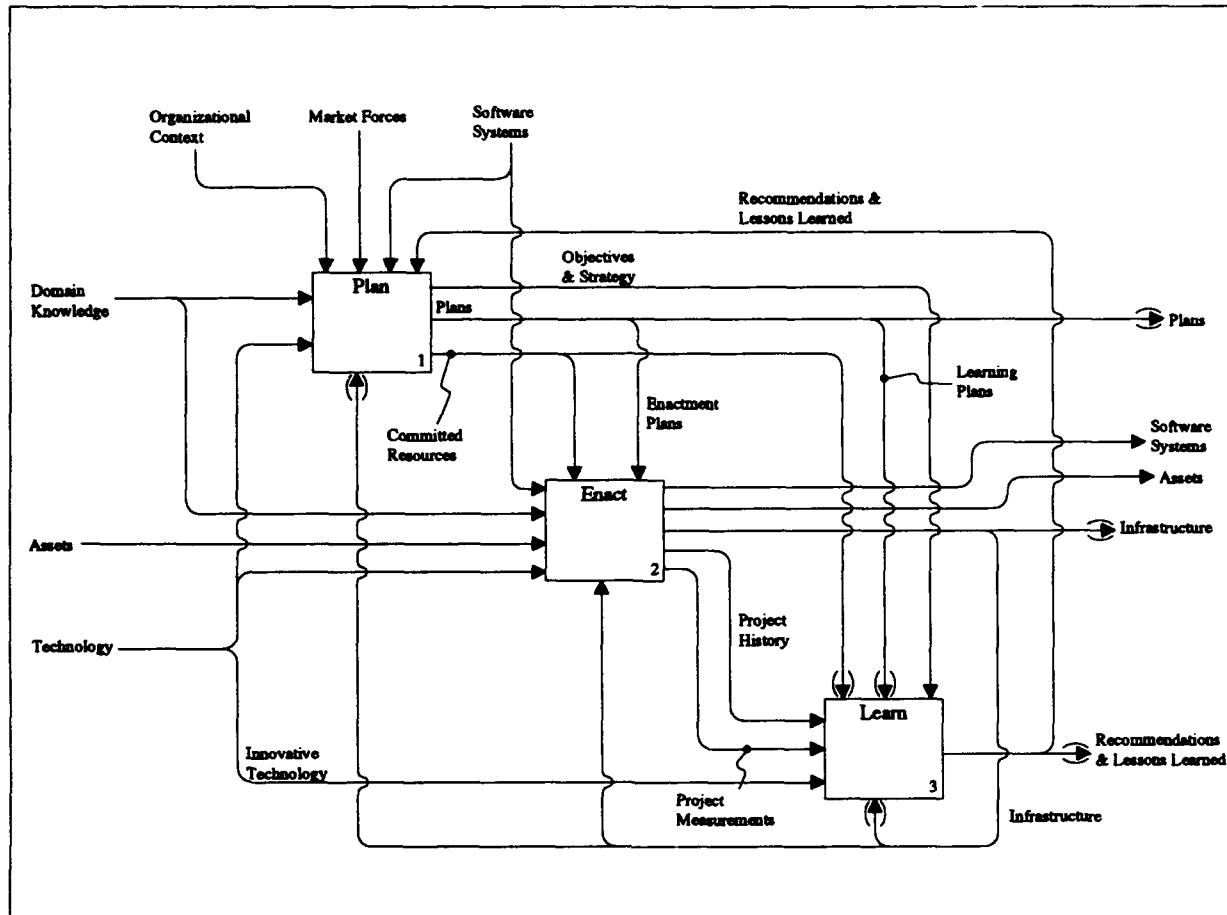


Figure 9: Perform Reuse-Based Software Evolution IDEF₀ Diagram

- Plan Infrastructure
- Plan Projects
- Manage Projects
- Implement Infrastructure
- Observe Projects
- Evaluate Projects
- Explore Innovation
- Recommend Enhancements

These are the same categories that were shown on the Organization Management Circle Chart in Figure 2. These categories and the corresponding activities and data flow will be discussed in further detail in this section.

Another view of the activities, with flow of data between them, can be seen in the IDEF₀ diagrams. Figure 9 shows the A0 level IDEF₀ Diagram for ROSE. This diagram shows that the top level **Perform Reuse-Oriented Software Engineering (ROSE)** Process is broken hierarchically into the **Plan**, **Enact**, and **Learn** Processes of the Organization Management submodel.

As described in greater detail in [12], the Plan-Enact-Learn structure of Organization Management forms a cyclic pattern of activity addressing the establishment and continual improvement of reuse-oriented processes and products within an organization by emphasizing learning as an institutional mechanism for change. Although the Plan Process occurs first in the IDEF₀ Diagram, it is important to note that ROSE users may tailor Organization Management cycles to begin with Enact (prototyping, exploratory style) or Learn (emphasizing research and empirical data collection from existing practice). Also, although the overall pattern is cyclic, there may be considerable concurrent activity among the processes. For example, while projects are being enacted (Enact Process), observation and learning about the projects and infrastructure may be underway (Learn Process), and planning activities may have begun for the ensuing cycle or other projects (Plan Process).

The **Plan** process family addresses both long- and short-term planning for reuse-based software-engineering within a line of business organization. Inputs to the Plan Process include **Domain Knowledge** that can be imparted in a variety of ways to provide information about the domain; and **Technology** that can contribute to the infrastructure and can be applied to automate processes. The Plan Process is controlled by **Organizational Context** that identifies the business strategies, policies and procedures, expertise, technologies, cultural legacies, etc., of an organization; **Market Forces** that identify significant new market trends, competitive developments, new technologies, emerging standards, and other factors that impact perception of marketplace needs; and **Software Systems** in encompassing domains of interest that can impart legacy knowledge about those domains. The Organization's **Infrastructure** can be used as a mechanism to automate Plan Process; it may also influence infrastructure planning for the engineering projects. Outputs of Organization Management include **Plans** for carrying out the Enact and Learn Processes; **Objectives and Strategy** for the organization which are compared to actual practices in the Learn Process; and **Committed Resources** to be used to carry out the plans.

The **Enact** process family manages active projects, regulates overall project performance, and ensures that an **Infrastructure** sufficient to meet the needs of the projects is established and maintained. Individual projects are embedded inside the Enact Process hierarchically. These projects include Domain Engineering projects, which transform **Domain Knowledge** and **Software Systems** within the domain into domain **Assets** (domain models, domain architectures, components, and application generators). Asset Management projects produce reuse libraries containing managed assets. Application Engineering projects transform requirements into new **Software Systems**. All projects provide **Project Measurements** and **Project History** to the Learn Process.

The **Learn** process family assesses the overall success of the strategies and plans put into

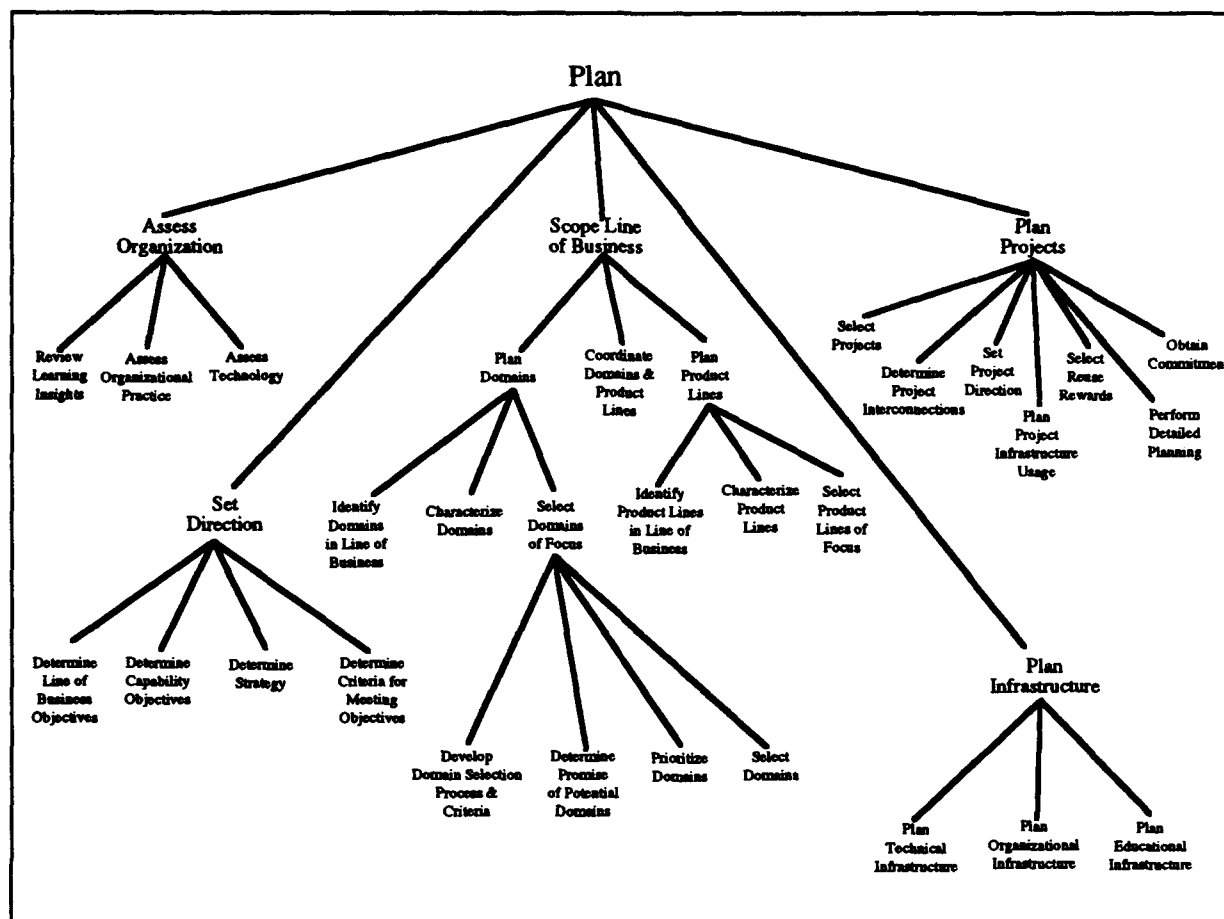


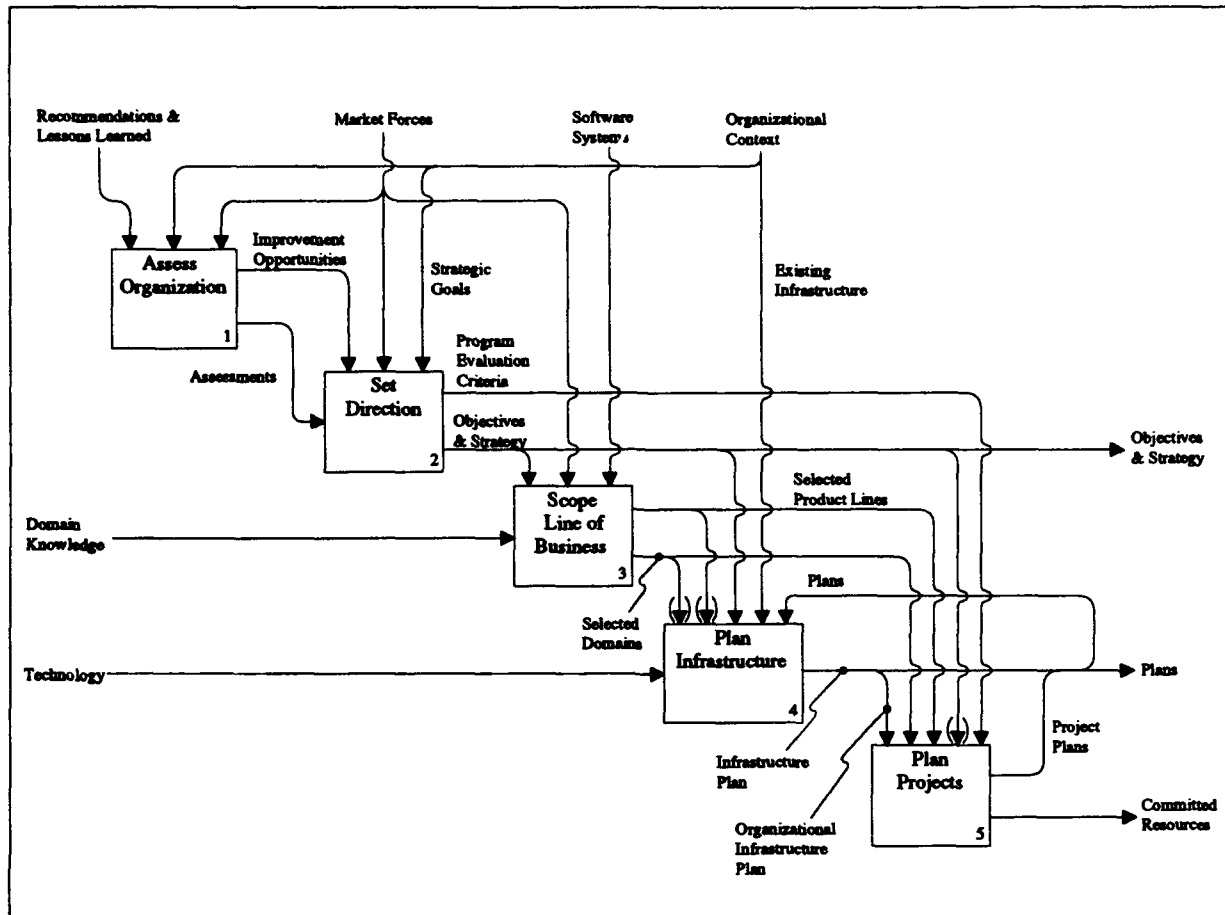
Figure 10: Organization Management Planning Process Abstraction Hierarchy

place by the Plan process, comparing **Project Measurement** and **Project History** data against project and program **Objectives** and **Strategy**, to both improve the quality of the plans and enhance the overall capabilities of the organization. The results of the Learn Process are fed back to the Plan Process in the form of **Recommendations** and **Lessons Learned** for the next reuse program cycle. The Learn Process also evaluates **Innovative Technology** from outside the organization to assess whether the technology can be used to improve the organization.

The following subsections describe the Plan, Enact, and Learn processes within Organization Management in greater detail.

3.3.1 Plan

The Plan process family consists of both long- and short-term planning processes for reuse-based software engineering. Long-term planning includes strategic assessments of how best to apply organization resources to initiate and sustain reuse-based development in accordance with overall business objectives. Integral to long-term planning is the determination

Figure 11: Plan IDEF₀ Diagram

of the product lines and domains of focus for the organization. Short-term planning addresses topics such as specific project processes, project interconnections, project infrastructure requirements, and resource planning. For each subsequent cycle after the first one, planning also includes review of recommendations and lessons learned from previous cycles and modification of plans based on the recommendations. The process abstraction hierarchy diagram for the Plan process family within Organization Management is shown in Figure 10.

Figure 11 contains an IDEF₀ Diagram for the Organization Planning process family. As shown in the figure, the Plan process consists of the following five process categories:

- **Assess Organization** processes characterize the current state of practice of the organization. Another function of assessment is to review the Recommendations and Lessons Learned from previous cycles and determine specific improvements to be made based on these recommendations.
- **Set Direction** processes compare assessment results to the strategic goals of the organization to define objectives and a strategy for reaching those objectives. An improvement strategy is also defined.

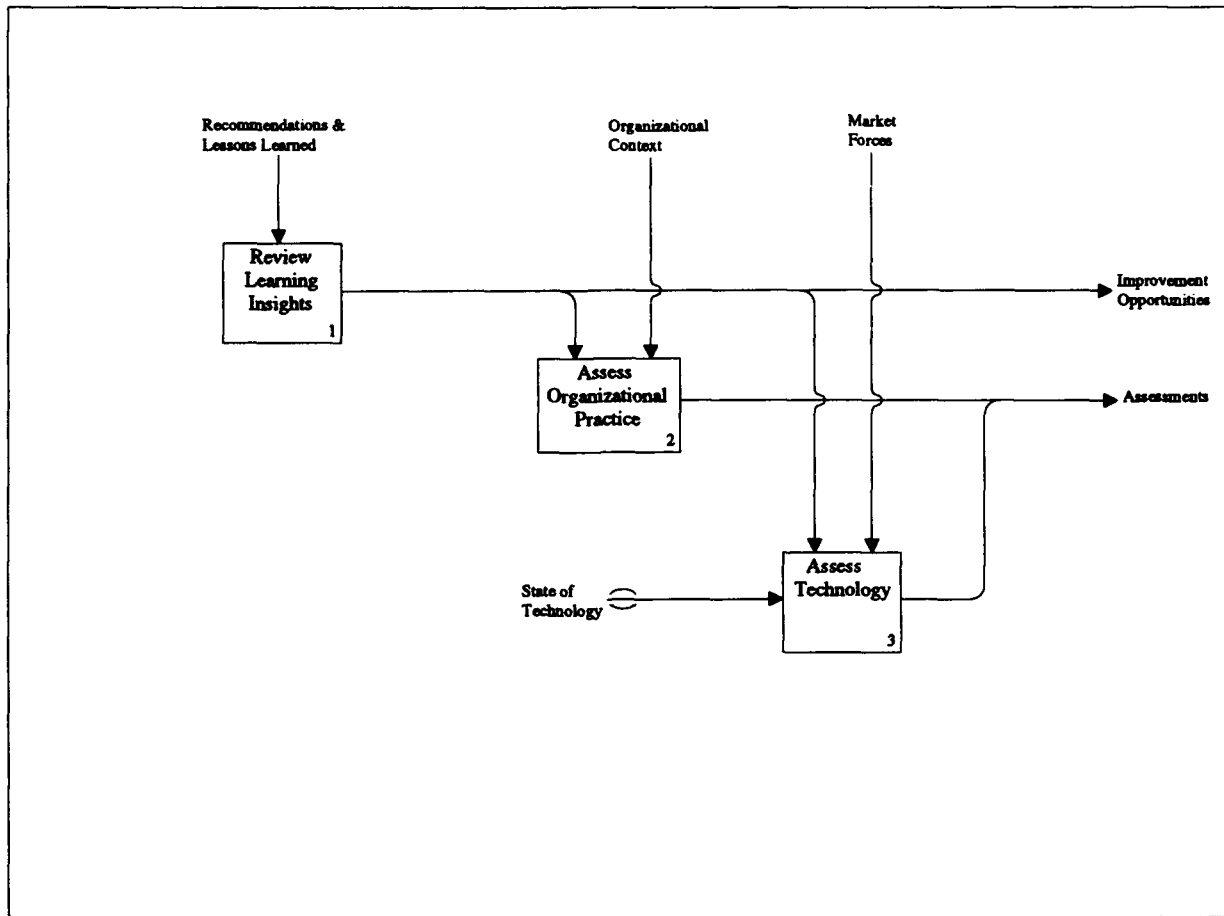


Figure 12: Assess Organization IDEF₀ Diagram

- **Scope Line of Business** processes determine the product lines that the organization will focus on producing and the domains of focus that will be instrumental in developing those product lines.
- **Plan Infrastructure** processes assess common organizational, technical, and educational infrastructure needs for all projects within the organization.
- The **Plan Projects** processes determine the Domain Engineering, Asset Management, and Application Engineering projects that will be performed by the organization. Detailed planning of the interactions among specific projects within the organization is performed, as is planning of the interactions between those projects and external asset producer, broker, and consumer organizations, as appropriate.

3.3.1.1 Assess Organization

The Assess Organization Process Category includes processes for assessing the current state of the organization and technology and for reviewing recommendations and lessons learned.

The Assess Organization IDEF₀ diagram is shown in Figure 12. Assess Organization includes the Review Learning Insights, Assess Organizational Practice, and Assess Technology processes.

The **Review Learning Insights** process reviews feedback from the Learn Process Family. The feedback contains information about state of practice within the organization, specific recommendations (with rationale) based on assessment of current practice, lessons learned, and new discoveries and innovations. These inputs from Learn are only recommendations. The Assessment activities evaluate the recommendations and incorporate them into an overall assessment of needs and potential solutions or improvements.

The **Assess Organizational Practice** process characterizes the current state of practice within the organization's scope of planning and the readiness of the organization as a whole for practicing reuse-based, process-driven, technology-supported software engineering. These assessments include assessments of the process capabilities, reuse capabilities, and infrastructure capabilities of the organization. This process may also assess the organization's strategies and competitive position within its line of business, its core competencies and overall application and domain expertise, and so on.

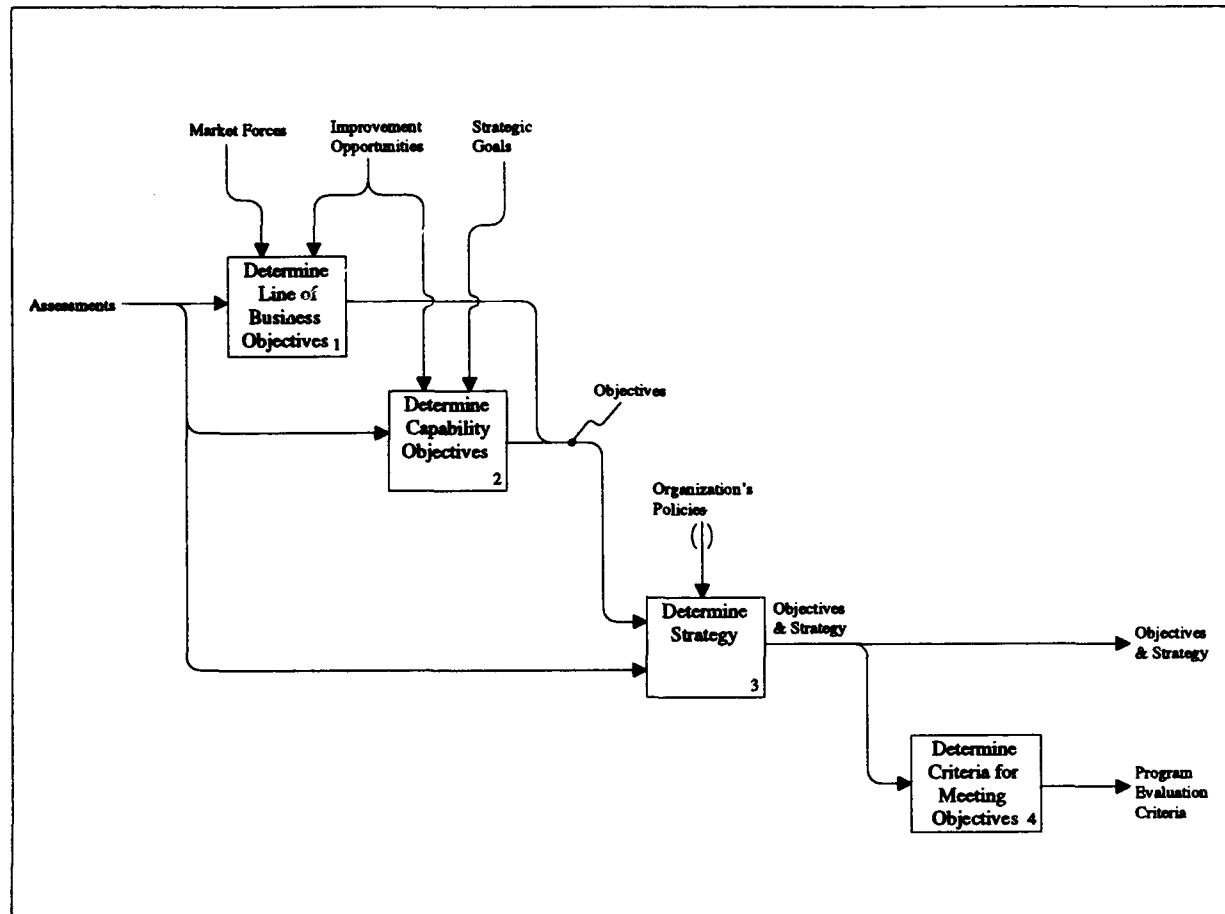
The **Assess Technology** process assesses the technology and expertise available both internal and external to the organization. This assessment includes determining new technology available in the process, reuse, and software engineering environment areas. A Market Factors Assessment may also be performed to determine competitive challenges, new technologies, emerging standards, developments within application domains, and other external market factors.

3.3.1.2 Set Direction

The Set Direction Process Category compares the results of assessment with strategic goals established for the organization as a whole in order to define specific objectives, strategies for reaching those objectives in the organizational context, and criteria for evaluating how successfully the objectives have been met. The Set Direction IDEF₀ diagram is shown in Figure 13. Set Direction includes the Determine Line of Business Objectives, Determine Capability Objectives, Determine Strategy, and Determine Criteria for Meeting Objectives processes.

The **Determine Line of Business Objectives** process develops an overall vision and mission for the organization. This vision and mission should be based directly on the Organizational Assessments and Improvement Opportunities determined in the Assess Organization processes.

The **Determine Capability Objectives** process determines the organization's reuse, process, and technology objectives based on the technology assessments developed in Assess Organization. For each of these areas the organization should develop a vision and mission.

Figure 13: Set Direction IDEF₀ Diagram

The **Determine Strategy** process develops more detailed objectives and a strategy for reaching the organization's vision and mission. It is based on the overall vision and mission and the vision and mission for the reuse, process and technology areas. The strategy should balance the needs of these different areas. The objectives and strategy developed here will impact Line of Business scoping and will be used to plan and guide the Domain Engineering, Asset Management, and Application Engineering projects employed within the organization.

The **Determine Criteria for Meeting Objectives** process develops measurement criteria that can be used to assess whether the organization met its objectives and strategy. These criteria should be specific enough to allow qualitative and quantitative measurement of progress against the objectives.

3.3.1.3 Scope Line of Business

The Scope Line of Business process category involves the selection of product lines and domains for the organization to focus on, within the organization's line of business. As mentioned previously, the overall organizational context of ROSE is a single line of business.

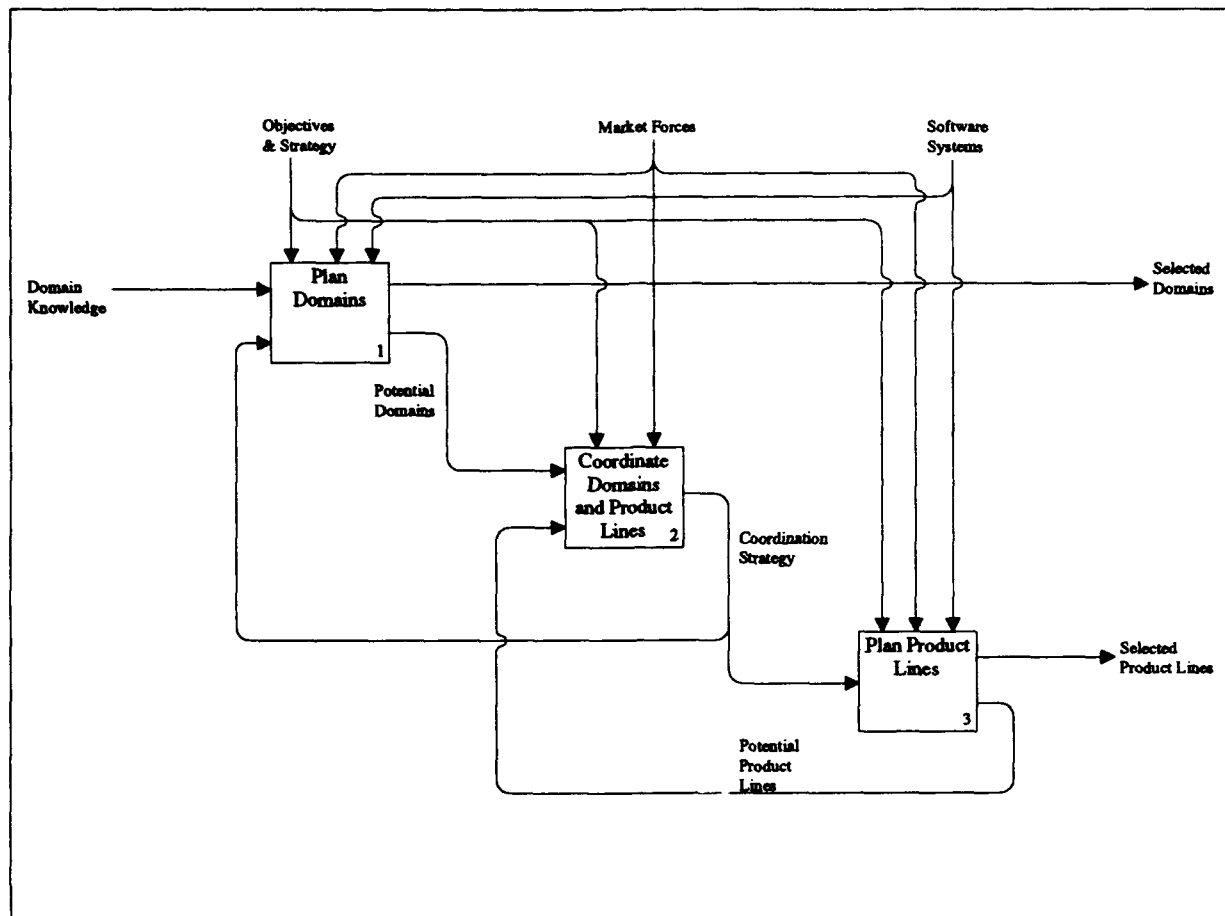
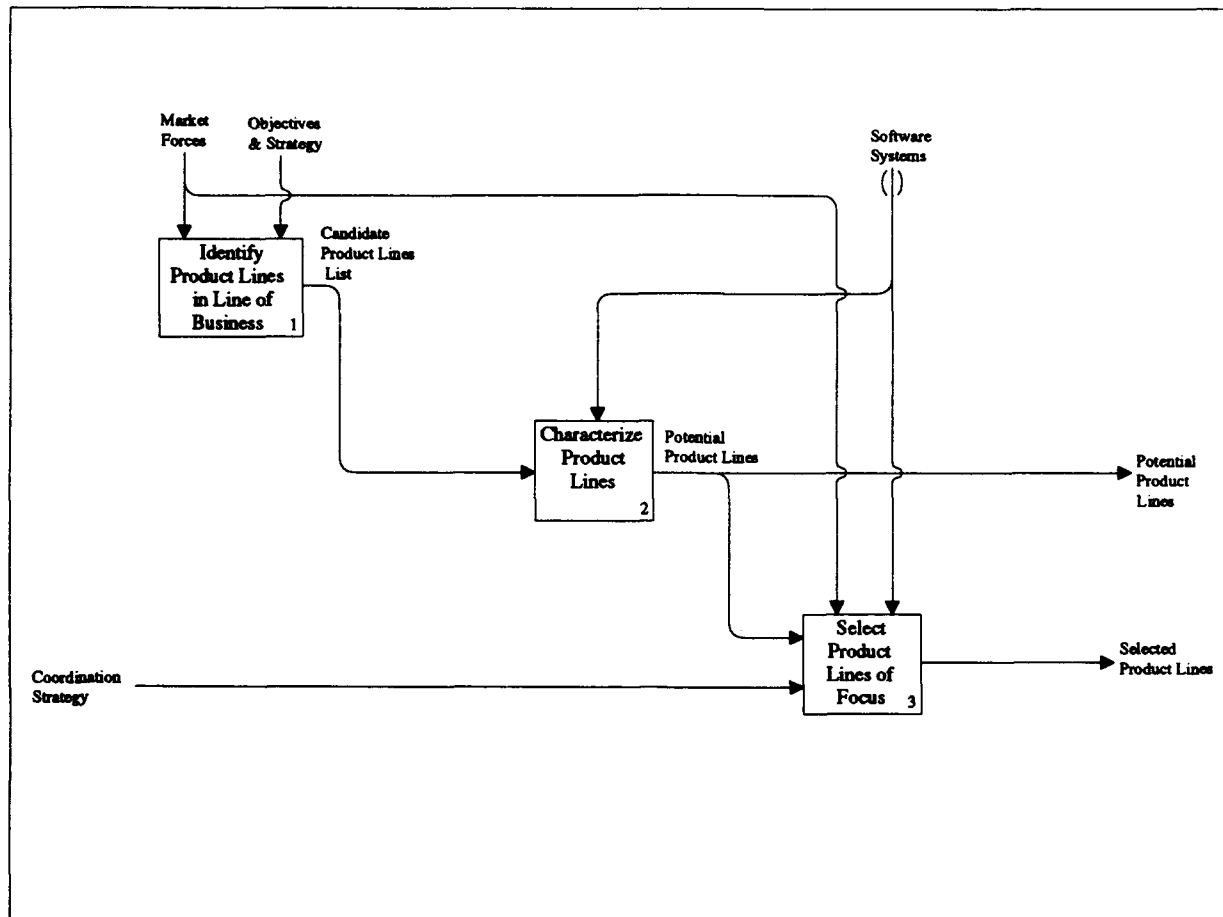


Figure 14: Scope Line of Business IDEF₀ Diagram

This means that the Organization Management Plan, Enact and Learn process families act within the context of this line of business. The line of business is generally determined by the organization's parent or by high-level planning processes that are not addressed in ROSE.

Within Organization Management, functional domains must be selected as focuses for Domain Engineering projects. Similarly, product lines are selected as focuses for potential Application Engineering projects, based on customers within the line of business. These product lines are used as the basis for assessing potential domains of interest to determine the domains of focus for the current cycle of Domain Engineering projects.

The Scope Line of Business IDEF₀ diagram is shown in Figure 14. Scope Line of Business includes the Coordinate Domains and Product Lines, Plan Product Lines, and Plan Domains processes.

Figure 15: Plan Product Lines IDEF₀ Diagram

Coordinate Domains and Product Lines

The Coordinate Domains and Product Lines process is a critical process which ensures that the planned product lines and domains of focus are coordinated and complement each other. Within the context of the organization's Objectives and Strategy and external Market Forces, potential product lines and potential domains of interest are compared to ensure a good fit between the two. The coordination activity works in parallel to product line and domain planning, with all three activities ending when a good match is obtained between domains and products.

Plan Product Lines

The Plan Product Lines process addresses the identification and characterization of potential product lines from the organization's business interests and current and potential customer base. Once these product lines are identified, product lines of focus are selected. These product lines are used along with potential domains to determine domains of focus for the current cycle of Domain Engineering projects. The focus of Application Engineering

projects within the organization depends on whether the organization is in a commercial or contracting environment. In a commercial environment, Application Engineering projects may be determined directly from the most promising selected product lines. In a contracting environment, the selected product lines will generally determine the proposal requests that the organization will pursue. In this latter environment, Application Engineering projects performed depend on which contracts are awarded by the customers.

The Plan Product Lines IDEF₀ diagram is shown in Figure 15. Plan Product Lines includes the **Identify Product Lines in Line of Business**, **Characterize Product Lines**, and **Select Product Lines of Focus** processes.

The **Identify Product Lines in Line of Business** process is concerned with the development of a list of all potential product lines within the organization's line of business. These product lines are targets based on marketing opportunities and current and potential customer base. At this point it is not necessary to be selective. All relevant candidate product lines should be identified. Interviewing experts in the line of business and inventorying current products can assist in product line identification.

The **Characterize Product Lines** process characterizes candidate product lines according to a set of criteria relevant to the organization. This characterization is used to assist in selecting product lines of focus.

The **Select Product Lines of Focus** process prioritizes from among the potential product lines and selects one or more product lines as a focus for product development.

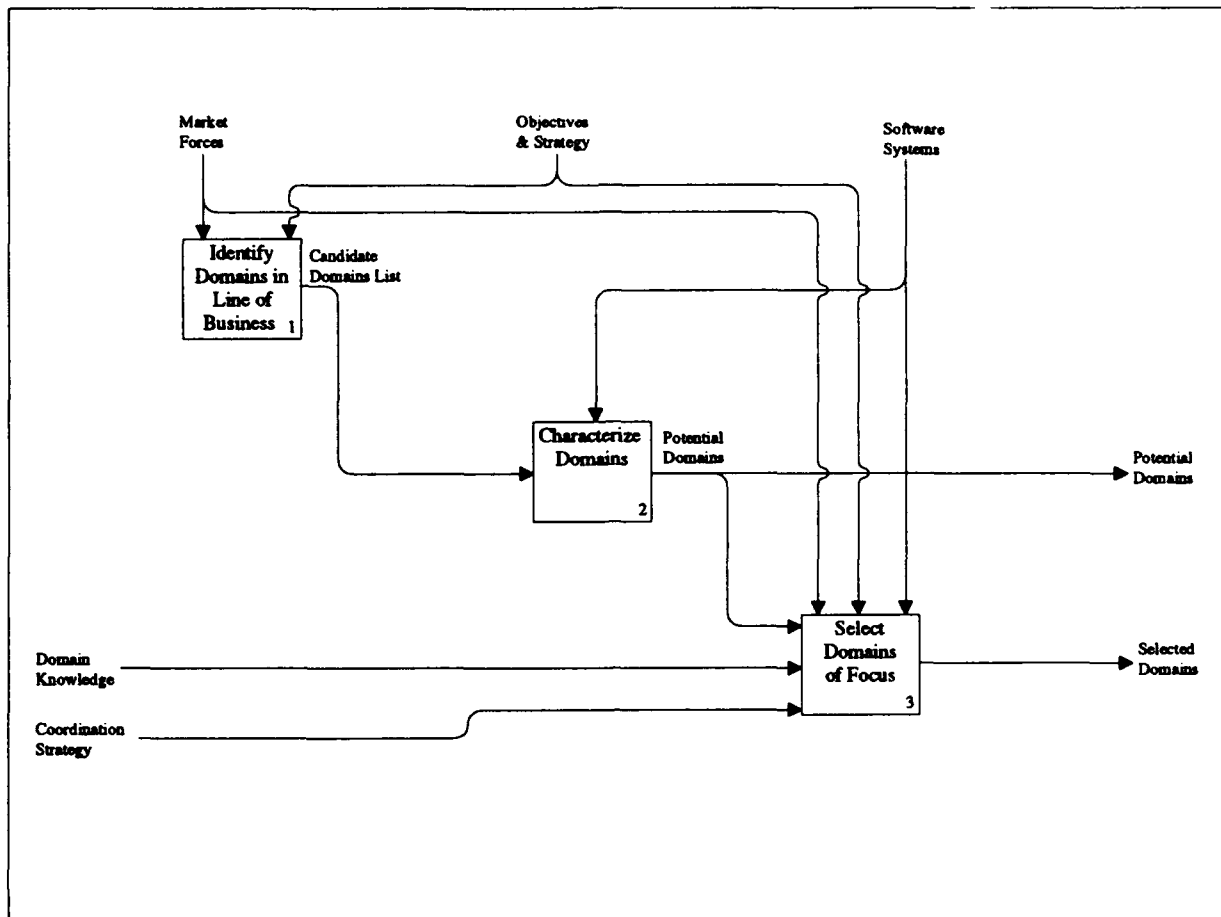
Plan Domains

The **Plan Domains** process addresses the identification and characterization of potential domains from the organization's business interests and existing legacy systems. Once these domains are identified, focus domains are identified for the current cycle of Domain Engineering projects. The Plan Domains IDEF₀ diagram is shown in Figure 16. Plan Domains includes the **Identify Domains in Lines of Business**, **Characterize Domains**, and **Select Domains of Focus** processes.

The **Identify Domains in Line of Business** process is concerned with developing a list of all potential domains within the organization's line of business. At this point it is not necessary to be selective. All relevant candidate domains should be identified. Interviewing experts in the line of business and inventorying current assets can assist in domain identification.

The **Characterize Domains** process characterizes candidate domains according to a set of criteria relevant to the organization. This characterization is used as a preliminary domain scoping. It is also used to assist in selecting domains of focus.

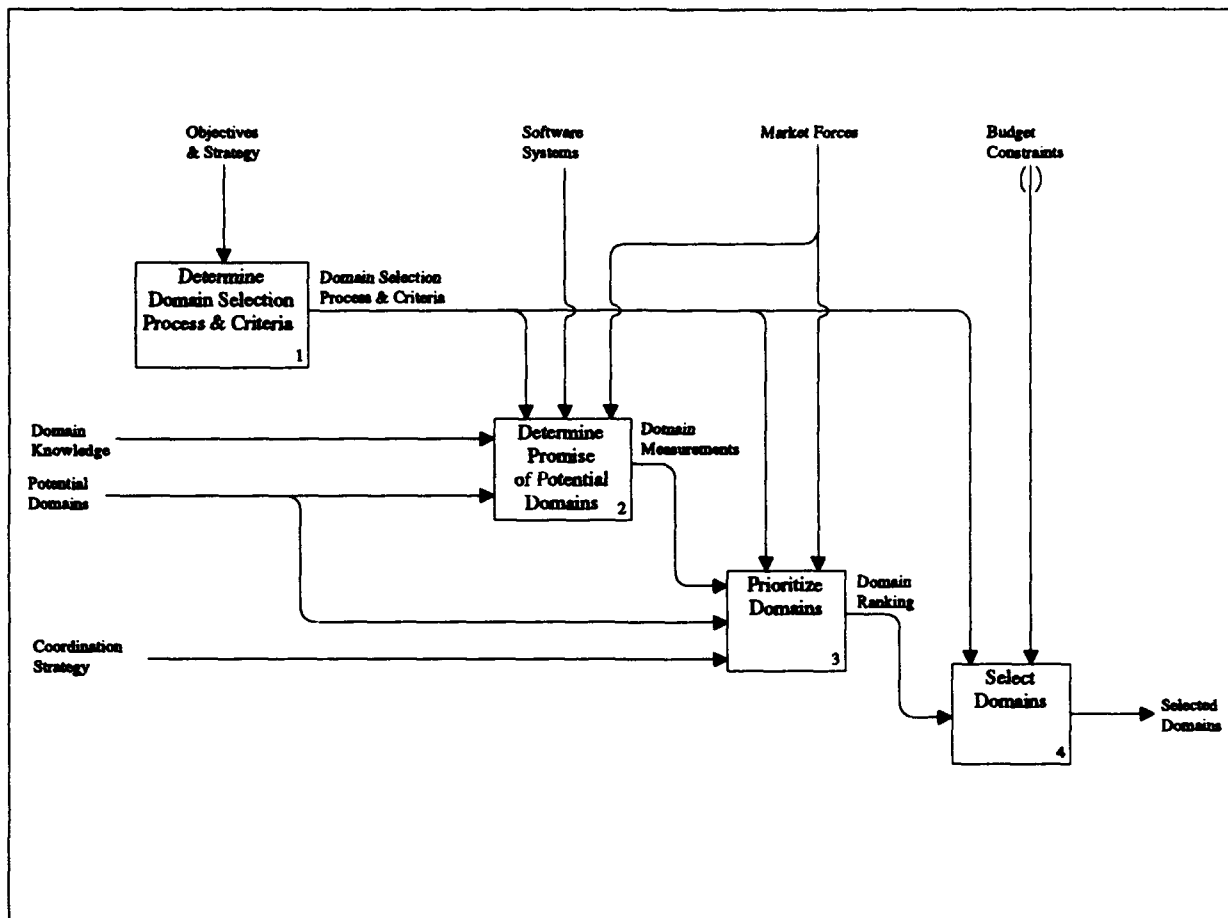
The **Select Domains of Focus** process prioritizes from among the potential domains stud-

Figure 16: Plan Domains IDEF₀ Diagram

ied and selects one or more domains for which specific reuse engineering projects are to be initiated. The IDEF₀ diagram for this process is shown in Figure 17. The process is composed of subprocesses to Determine Domain Selection Process and Criteria, Determine Promise of Potential Domains, Prioritize Domains, and Select Domains.

The first step in selecting domains of focus is to **Determine Domain Selection Process and Criteria**. The process and criteria used should be compatible with the Objectives and Strategy developed in the Set Direction Process Category. Documenting the process and criteria lays out the objectives of domain selection up front, so that they are agreed upon independent of the identified domains.

The **Determine Promise of Potential Domains** activity uses the domain characterizations and other domain knowledge to measure each domain against the selection criteria. In general the Characterize Domain process focuses on the general technical feasibility of a domain for reuse. Factors considered during the Determine Promise of Potential Domains activity include more strategic issues, such as the relative priority of the domain in the current environment of the organization, and economic factors such as return on investment. Timing factors that constrain the best choices of domains for the current program cycle may

Figure 17: Select Domain of Focus IDEF₀ Diagram

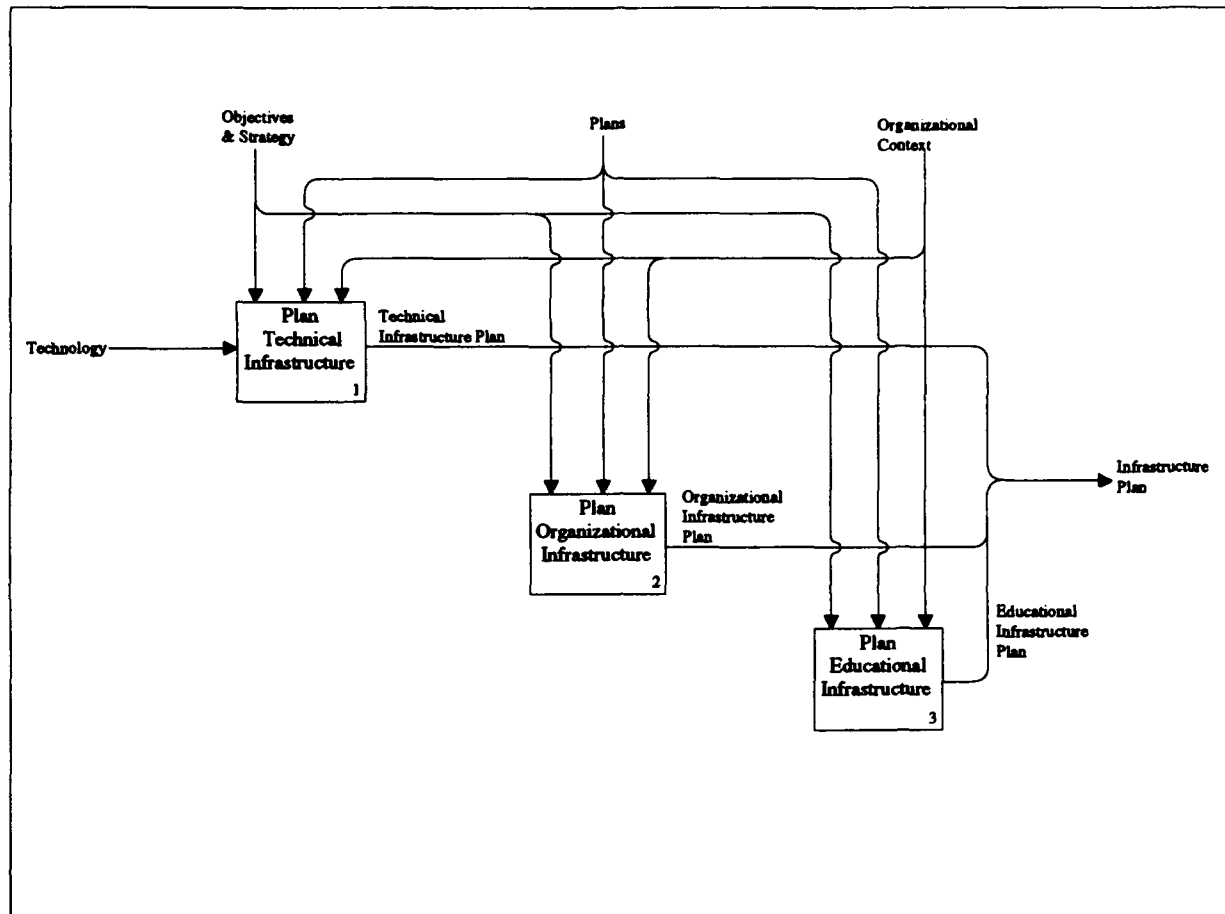
also be used.

The **Prioritize Domains** activity ranks all potential domains based on their promise. The objective of this activity is to obtain a ranked list with the domains with the highest payoff and lowest risk at the top.

The **Select Domains** activity selects the domains at the top of the ranking on which Domain Engineering projects will be pursued. The number of domains chosen will depend on the Budget Constraints of the organization.

3.3.1.4 Plan Infrastructure

The Plan Infrastructure Process Category assesses common infrastructure needs for all projects planned within the scope of the current program cycle. The infrastructure to support these common needs is developed as an overall program infrastructure to avoid duplication of effort on each of the projects. The Plan Infrastructure IDEF₀ diagram is shown in Figure 18. Plan Infrastructure includes the Plan Technical Infrastructure, Plan

Figure 18: Plan Infrastructure IDEF₀ Diagram

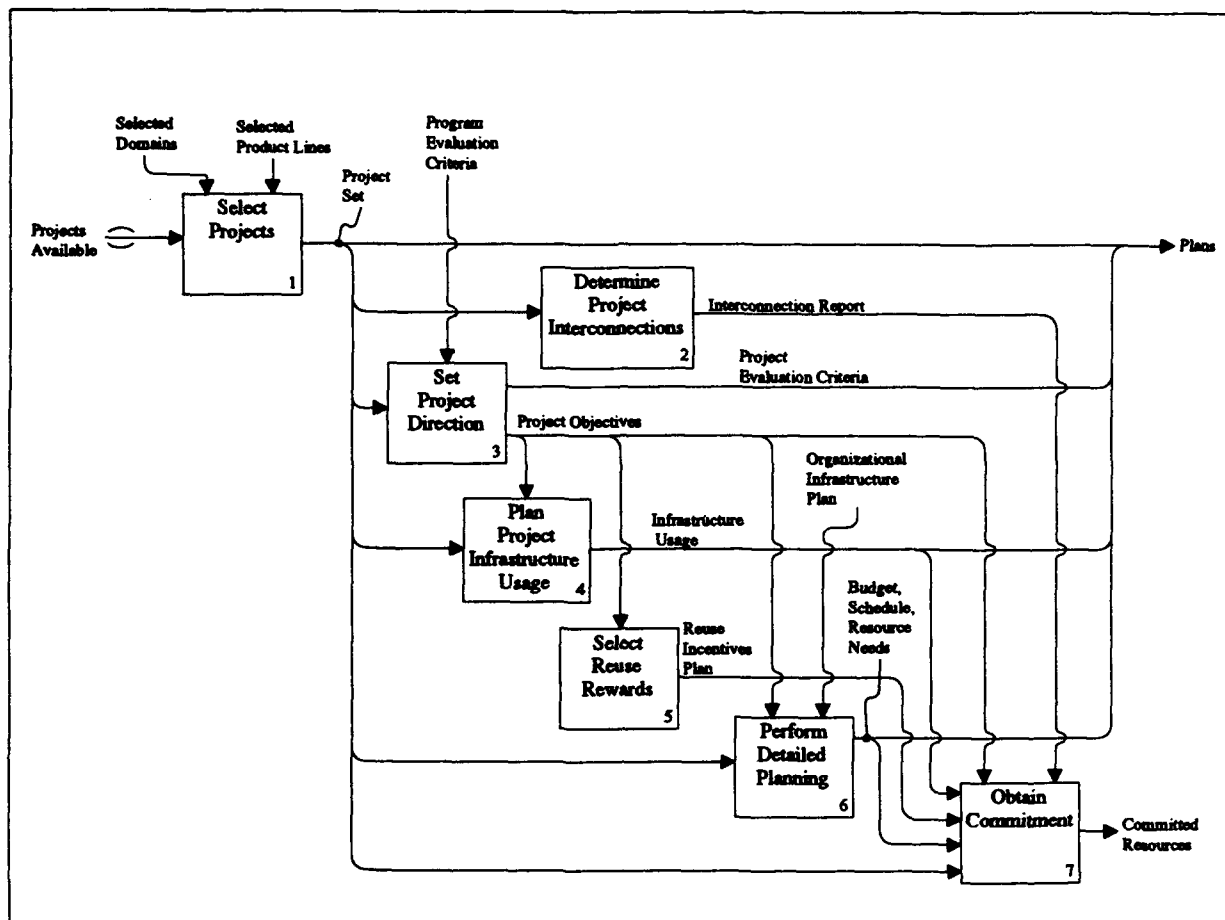
Organizational Infrastructure, and Plan Educational Infrastructure processes.

The **Plan Technical Infrastructure** process involves planning for automated tool support, including tools that encapsulate policy or decision-making, as well as tools to support domain modeling, library management, asset utilization, and project communication. This process also involves planning for organization-wide process models, particularly those incorporating reuse in a way that will address the needs of the reuse-based projects.

The **Plan Organizational Infrastructure** process involves planning coordinated team activities, funding models, staffing, incentive strategies, risk reduction measures, organizational policies and procedures, new project roles, establishment of specific task forces, transition teams and steering committees, and so forth.

The **Plan Educational Infrastructure** process involves planning development of necessary skills and capabilities on the part of individual workers.

3.3.1.5 Plan Projects

Figure 19: Plan Projects IDEF₀ Diagram

Processes in the Plan Projects Process Category perform high level planning of the Domain Engineering, Asset Management, and Application Engineering projects within the organization's scope. These processes also plan project interactions. Some iteration among Set Direction, Scope Line of Business, and Plan Projects processes may be required before the necessary commitment for the projects can be obtained. The Plan Projects IDEF₀ diagram is shown in Figure 19. Plan Projects includes the Select Projects, Determine Project Interconnections, Set Project Direction, Plan Project Infrastructure Usage, Select Reuse Rewards, Perform Detailed Planning, and Obtain Commitment processes.

The **Select Projects** process involves determining the Domain Engineering, Asset Management, and Application Engineering projects to pursue, based on product line and domain of focus planning. Asset Management may be centralized or split into several asset libraries.

Once individual projects are identified, dependencies and information flows between the designated Domain Engineering, Asset Management, and Application Engineering projects are made explicit in the **Determine Project Interconnections** process.

The **Set Project Direction** process involves determining the objectives and strategies for

the individual projects. This strategy is generally kept at a high level and is refined further within the Plan stage of each individual project.

The **Plan Project Infrastructure Usage** process involves allocating and tailoring the program's infrastructure to particular projects. Project management makes the final determination of infrastructure usage on each project. This preliminary allocation of infrastructure capabilities indicates to project management which parts of the infrastructure could support the project processes.

The **Select Reuse Rewards** process involves the selection of reuse rewards to use as incentives to project staff to develop and use reusable assets. Without these reuse incentives, project members may invent from scratch instead of spending the time to look for assets to reuse. Since assets may still need tailoring, it may take more time to find, tailor and integrate assets than to invent. It can nevertheless be beneficial from the overall organization perspective to reuse such assets, since testing and maintenance is spread over a number of projects instead of just one, and incentives can motivate such reuse.

The **Perform Detailed Planning** process consists of balancing the budget, schedule, staffing, and resource needs of the various projects.

The final stage in Reuse Planning is to **Obtain Commitment**, based on detailed planning of the various reuse projects' budgets, schedules, resource needs, and potential payoff. Commitment should be obtained from both higher level management and the technical staff members whose buy-in will ultimately determine whether or not reuse practice is really improved by the program.

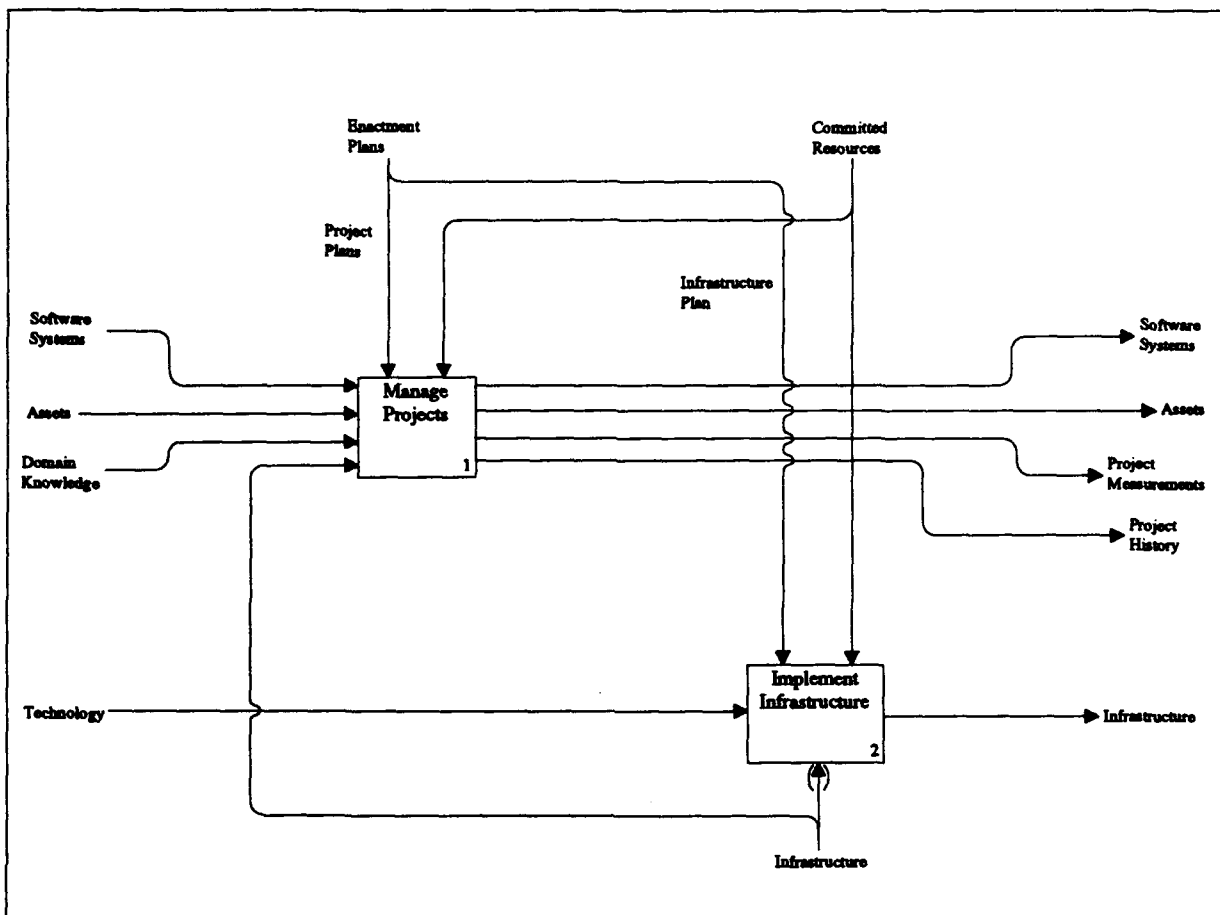
3.3.2 Enact

The Enact Processes manage the interactions between projects and ensure that an infrastructure sufficient to meet the needs of the projects is established and maintained.

Figure 20 contains an IDEF₀ diagram for the Organization Enactment process family. As shown in the figure, the Enact process family consists of the following two process categories:

- **Manage Projects** processes manage project interactions, including allocating resources among active projects, initiating and retiring projects, coordinating between projects, and monitoring interactions to ensure that planned project interconnections are utilized.
- **Implement Infrastructure** processes develop and evolve the planned infrastructure to support project needs.

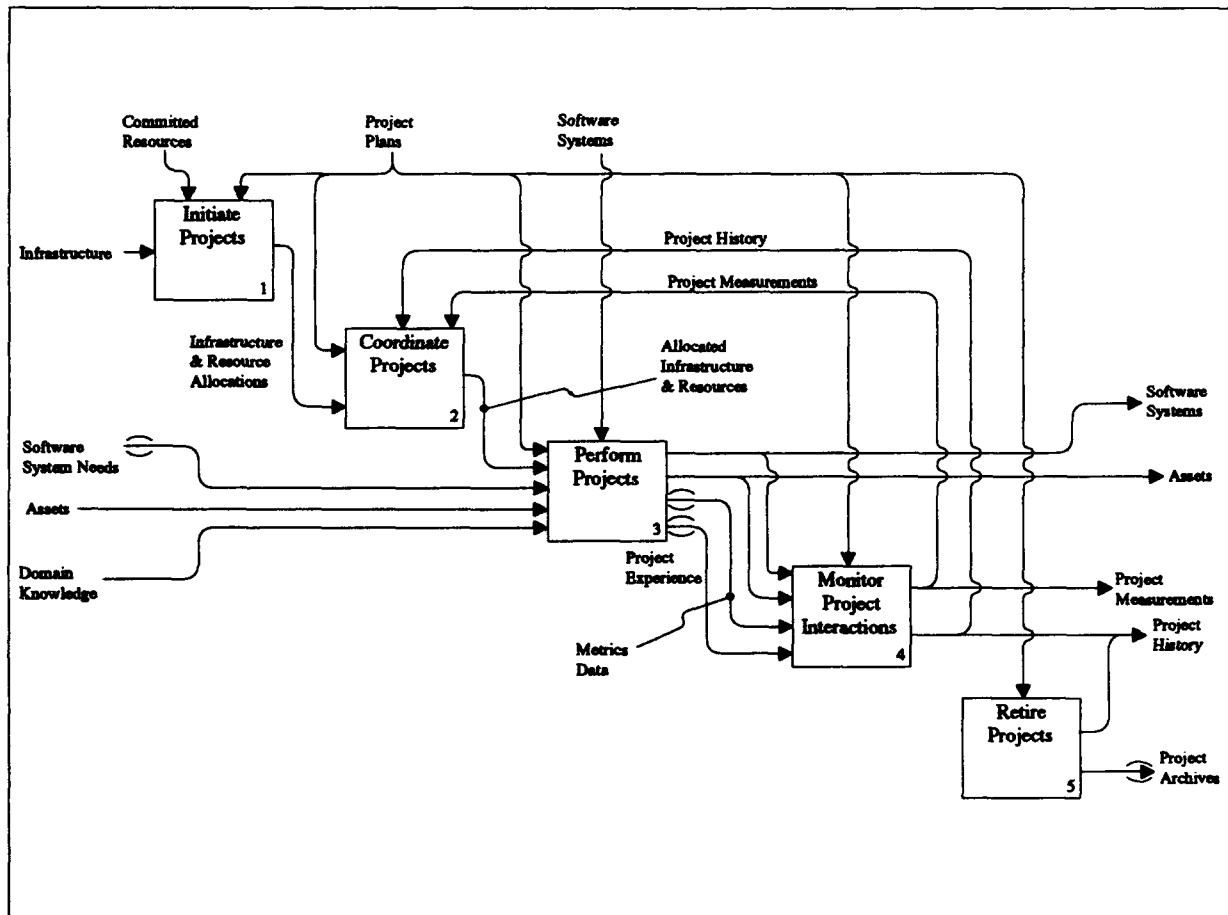
3.3.2.1 Manage Projects

Figure 20: Enact IDEF₀ Diagram

The processes within the Manage Projects Process Category regulate the overall performance on the projects within the organization, with a focus on managing the interactions among the projects and with external organizations. The Manage Projects IDEF₀ diagram is shown in Figure 21. Manage Projects includes Initiate Projects, Coordinate Projects, Perform Projects, Monitor Project Interactions, and Retire Projects.

The **Initiate Projects** process includes allocation and tailoring of reuse infrastructure capabilities to specific projects; allocation of project resources committed during Plan; staffing, training, and team formation for the specific organizational units involved in the projects; and application of incentives, funding strategies, and other relevant organizational policies to the projects. Once the project has been initiated, these project planning functions will be turned over to project management to be performed as part of controlling the project.

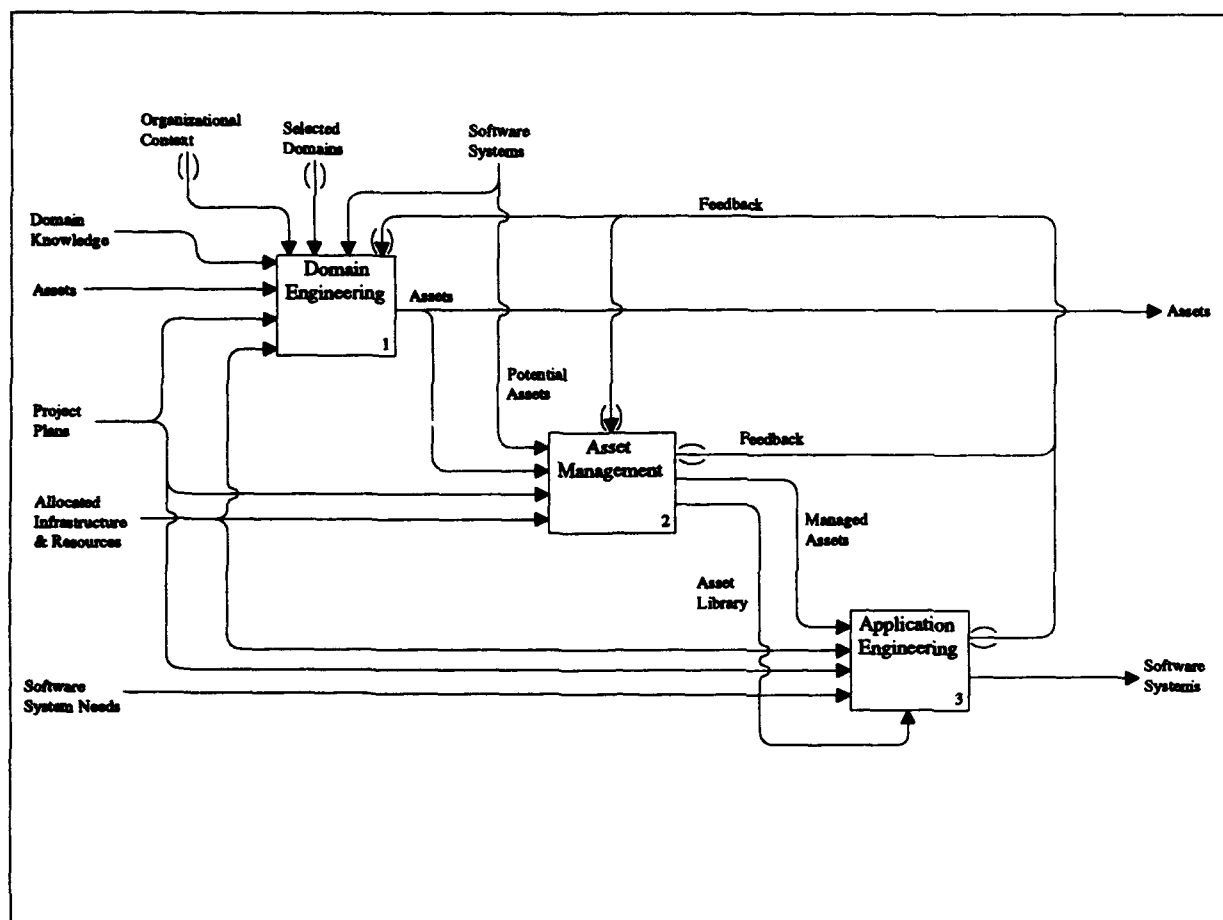
The **Coordinate Projects** process ensures that project interconnections are utilized to the benefit of all projects. Project History and Measurements are used to determine the extent of project interactions and to compare the interactions with planned interactions in the Project Plans. This coordination is necessary to determine whether the planned interactions take place in practice. Lack of interactions may indicate that staff on the projects need training

Figure 21: Manage Projects IDEF₀ Diagram

to take advantage of the interaction. Adjustments may be made to project interactions, assets created, products shaped within the product line, and external organizations with which projects interact. This includes adjustments to project teams, processes and policies, the software engineering environment, etc.

The **Perform Projects** process is where the Domain Engineering, Asset Management, and Application Engineering projects being enacted are “hooked in” to Organization Management and actually performed by individual staff members. Perform Projects also involves tactical decisions about *how* the projects should interact on a detailed, day-to-day basis, and how they should interact with external organizations.

Within Perform Projects one or more Domain Engineering, Asset Management, and Application Engineering projects are enacted. Figure 22 shows an example of the interactions that may occur between the individual projects when there is one Domain Engineering, one Asset Management, and one Application Engineering project being enacted. The primary data flows between these projects include the flow of **Assets** from Domain Engineering projects into **Asset Libraries** in Asset Management projects. **Managed Assets** within the **Asset Library** are utilized on Application Engineering projects. Feedback about the Assets

Figure 22: Perform Projects IDEF₀ Diagram

and Asset Libraries flows from Application Engineering to Asset Management and Domain Engineering, and from Asset Management to Domain Engineering.

The **Monitor Project Interactions** activities capture “learning-oriented” information about project interactions as they are performed. Some of this information provides feedback to the Coordinate Projects Process. Some of the same information may serve as input to the Learn Process Family. Both processes and products are measured. Process measurements assess conformance to planned processes and show the effectiveness of those processes. Product measurements assess conformance to and effectiveness of product line strategies.

The **Retire Projects** process can include termination procedures for projects, procedures and resources for archiving or maintaining essential project results, elimination of information needed solely to maintain project on an ongoing basis, and debriefing of experts on the knowledge gained during projects. Each project becomes part of an organization legacy that may be assessed in future program cycles

3.3.2.2 Implement Infrastructure

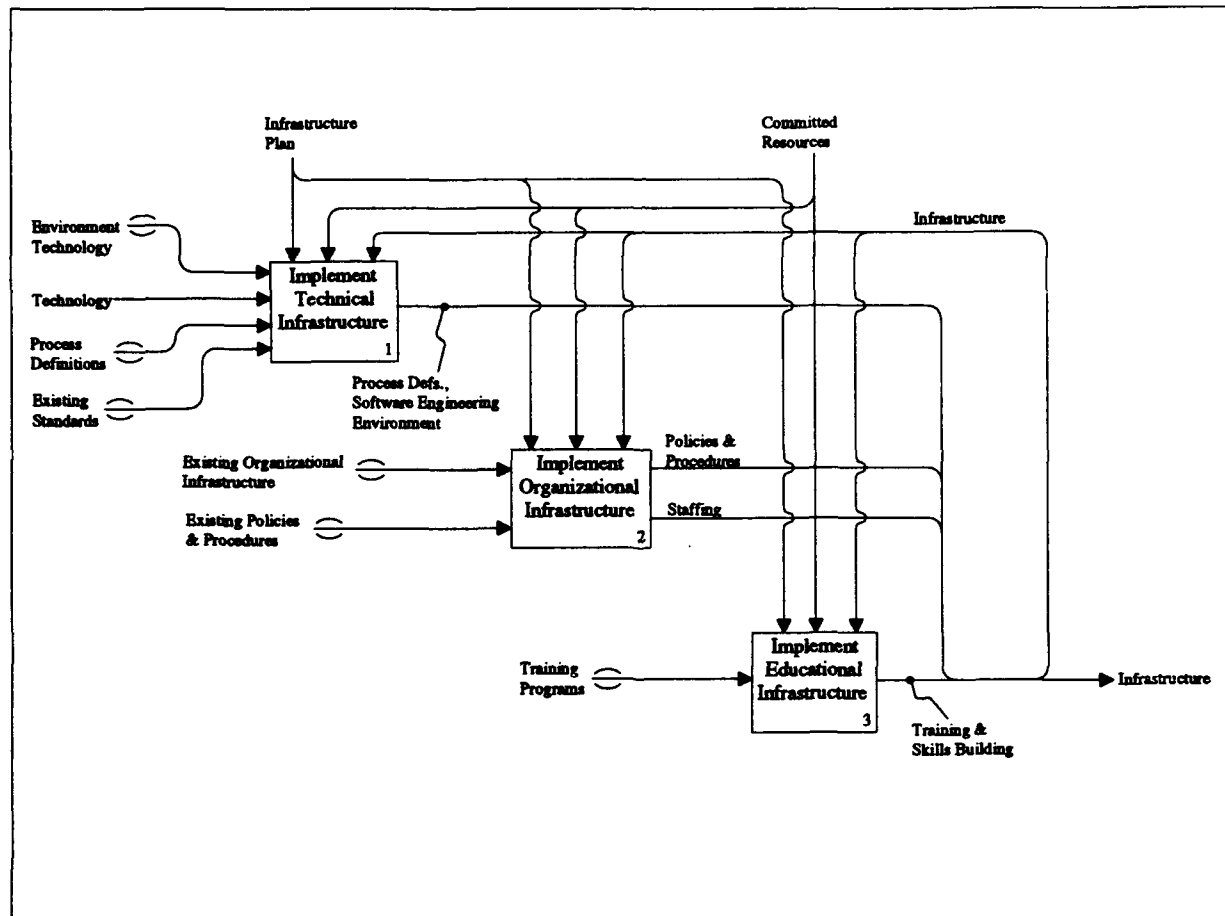


Figure 23: Implement Infrastructure IDEF₀ Diagram

Implement Infrastructure processes follow Infrastructure Plans to create a technical, organizational, and educational infrastructure that can be used by projects within the organization. Implementation of an infrastructure avoids the duplication of effort to implement the same processes, policies, software engineering environments, training, etc. on each individual project. It also allows the organization to have some control over the policies and procedures used by each project. In some organizations, projects must obtain approval to deviate from the infrastructure selected by the parent organization. If the infrastructure is managed in its own Asset Library, Infrastructure Asset Management activities are also needed.

The Implement Infrastructure IDEF₀ diagram is shown in Figure 23. Implement Infrastructure includes the Implement Technical Infrastructure, Implement Organizational Infrastructure, and Implement Educational Infrastructure processes.

Implement Technical Infrastructure includes development of software process descriptions, methods, and standards; tools and environments that can provide automated support for reuse processes; and computer hardware and other capital resources necessary for the reuse projects.

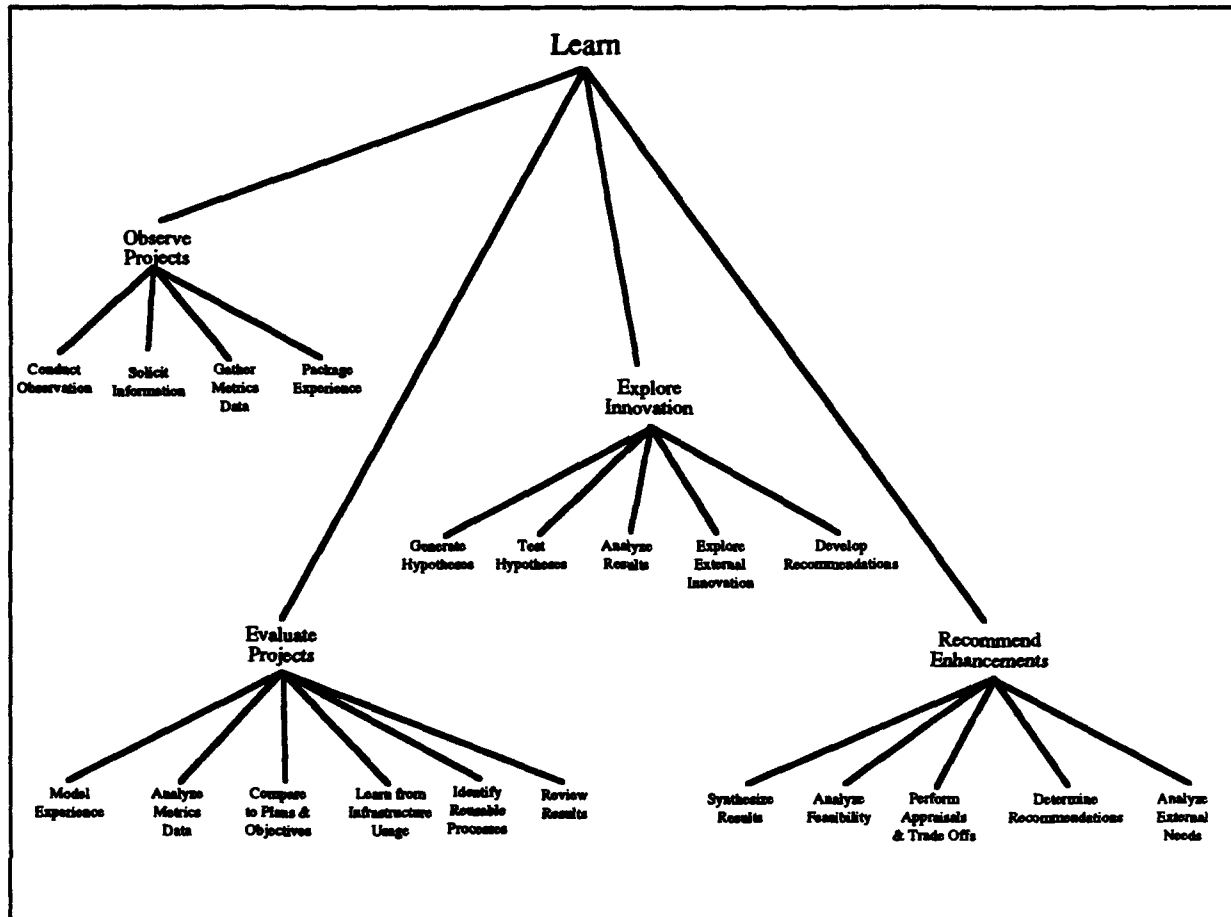
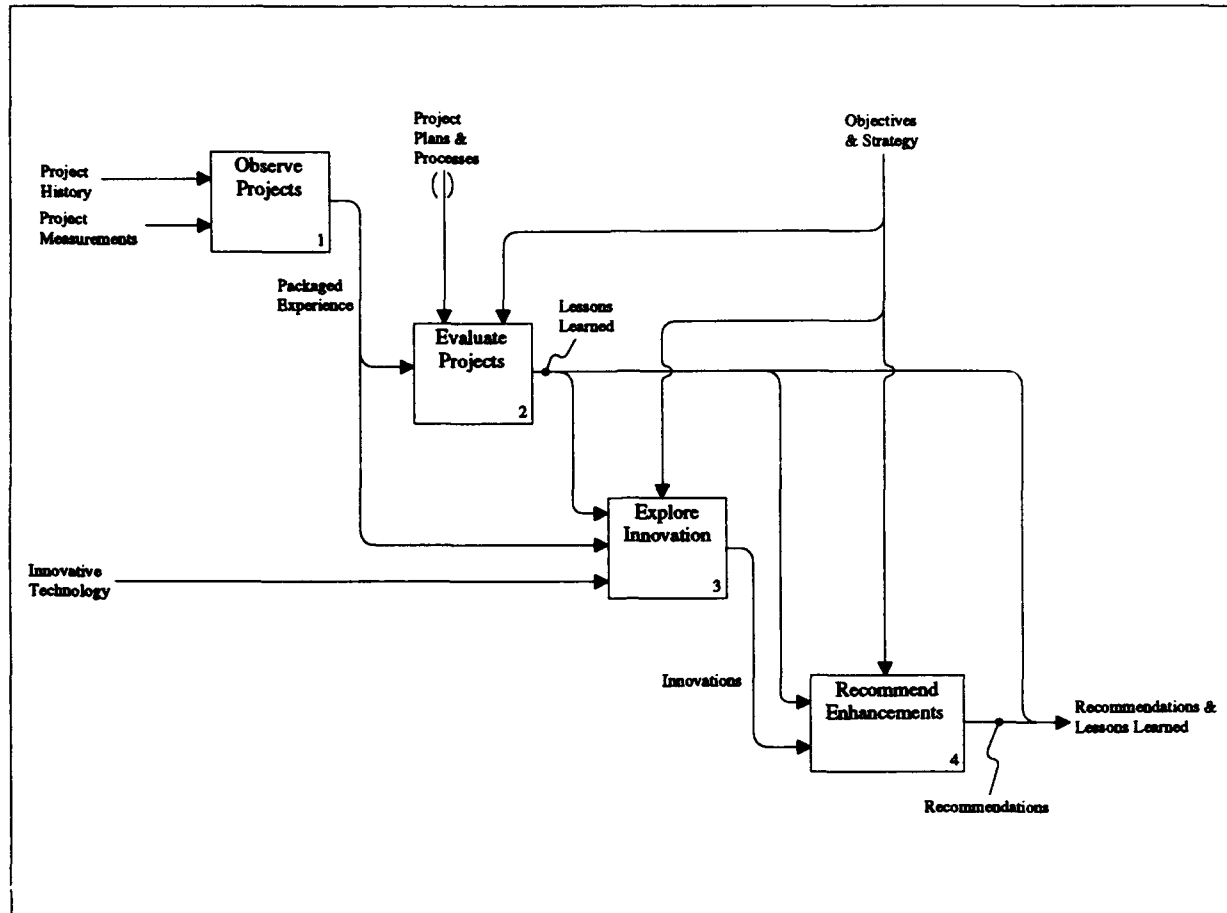


Figure 24: Organization Management Learning Process Abstraction Hierarchy

The **Implement Organizational Infrastructure** process includes selecting, tailoring, authorizing and communicating to developers the policies, procedures, and guidelines for various tasks. Implementing organizational infrastructure can also involve coordinating team activities; establishing, codifying, and promoting funding models, incentive strategies, and risk reduction measures; institutionalizing new roles within projects; and establishing specific task forces, transition teams, or steering committees.

The **Implement Educational Infrastructure** process includes the development of necessary capabilities and skills among individual workers. Personnel must be selected or hired to perform in the new roles. Training in reuse, process, and technology concepts and skills may be needed to varying degrees on each project, depending on the background of the workers assigned to the project, and the organization may thus need to establish an in-house training capability to accommodate those educational needs, or acquire the training from external sources.

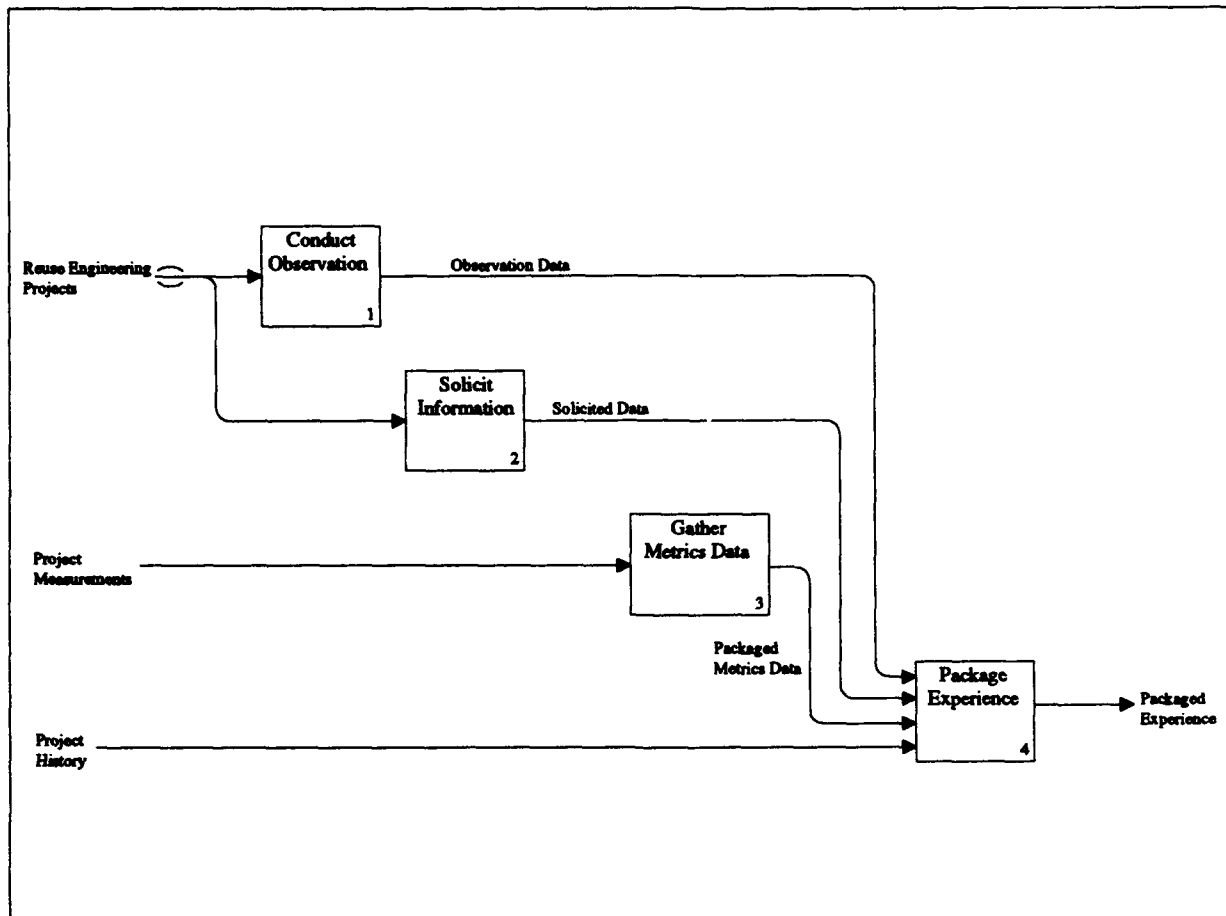
Figure 25: Learn IDEF₀ Diagram

3.3.3 Learn

Processes in the Organization Management Learn Process Family assess the overall success of the strategies and plans put into place by the Plan Process Family, comparing project results and experience against project and program objectives, to both improve the quality of the plans and enhance the overall reuse, process, and technology capabilities of the organization. The results of the Organization Management Learn Process Family are fed back to the Plan Process Family in the form of recommendations for the next program cycle. The process abstraction hierarchy diagram for the Learn process family within Organization Management is shown in Figure 24.

Figure 25 contains an IDEF₀ diagram for the Organization Management Learn Process Family. As shown in the figure, the Learn process family consists of the following four subprocesses:

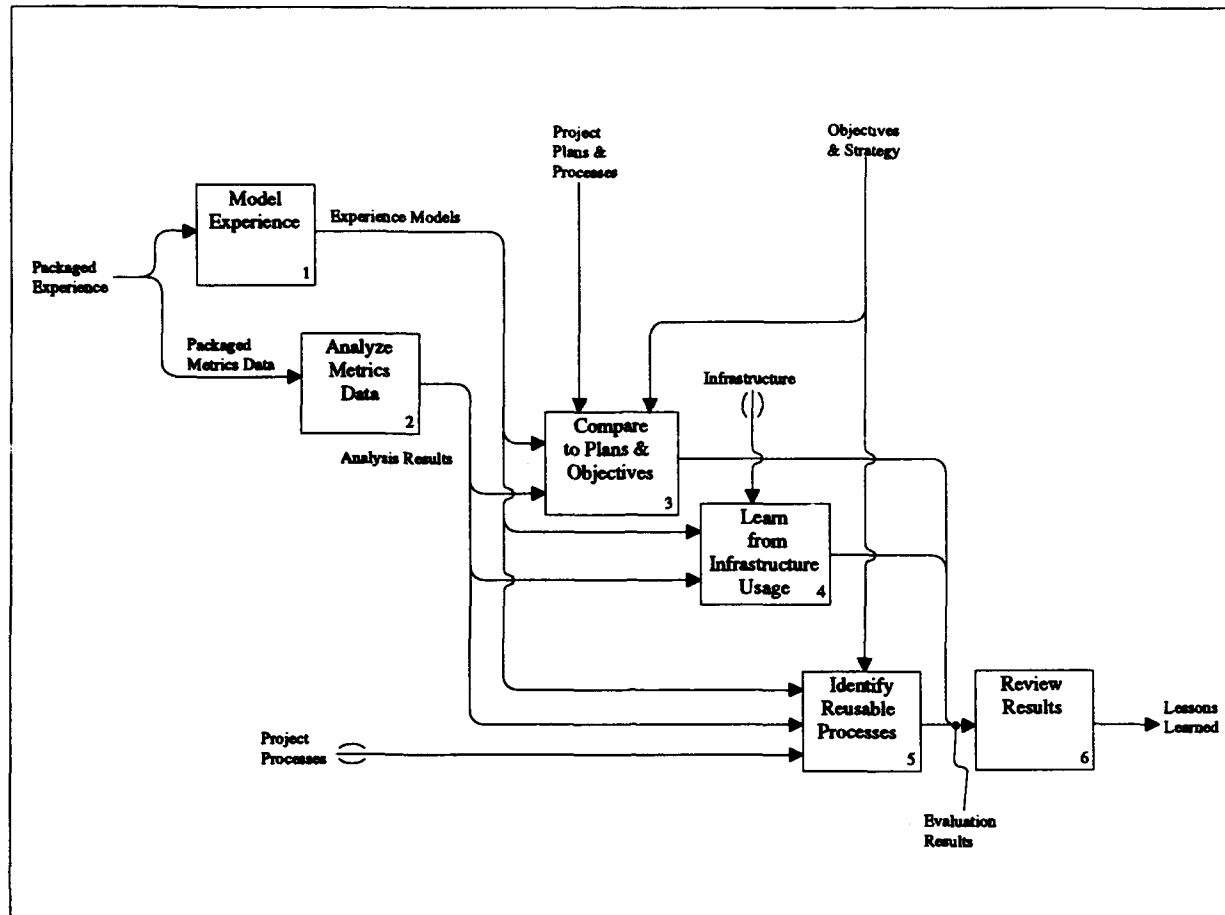
- **Observe Projects** processes gather Project Measurements and History data and package it into a form that can be evaluated.

Figure 26: Observe Projects IDEF₀ Diagram

- **Evaluate Projects** processes evaluate the gathered data and experiences against program objectives.
- **Explore Innovation** processes gather, generate, analyze, and test new ideas, discoveries, and innovations and make recommendations for using these concepts and capabilities to improve the organization.
- **Recommend Enhancements** processes collect and filter recommendations from Evaluate Projects processes and Explore Innovation processes back to Organization Planning.

3.3.3.1 Observe Projects

Process Observation processes gather information about the activities performed during Enactment that is considered potentially useful in assessing the performance of those activities relative to objectives and in generating recommendations for the next program cycle. Some of

Figure 27: Evaluate Projects IDEF₀ Diagram

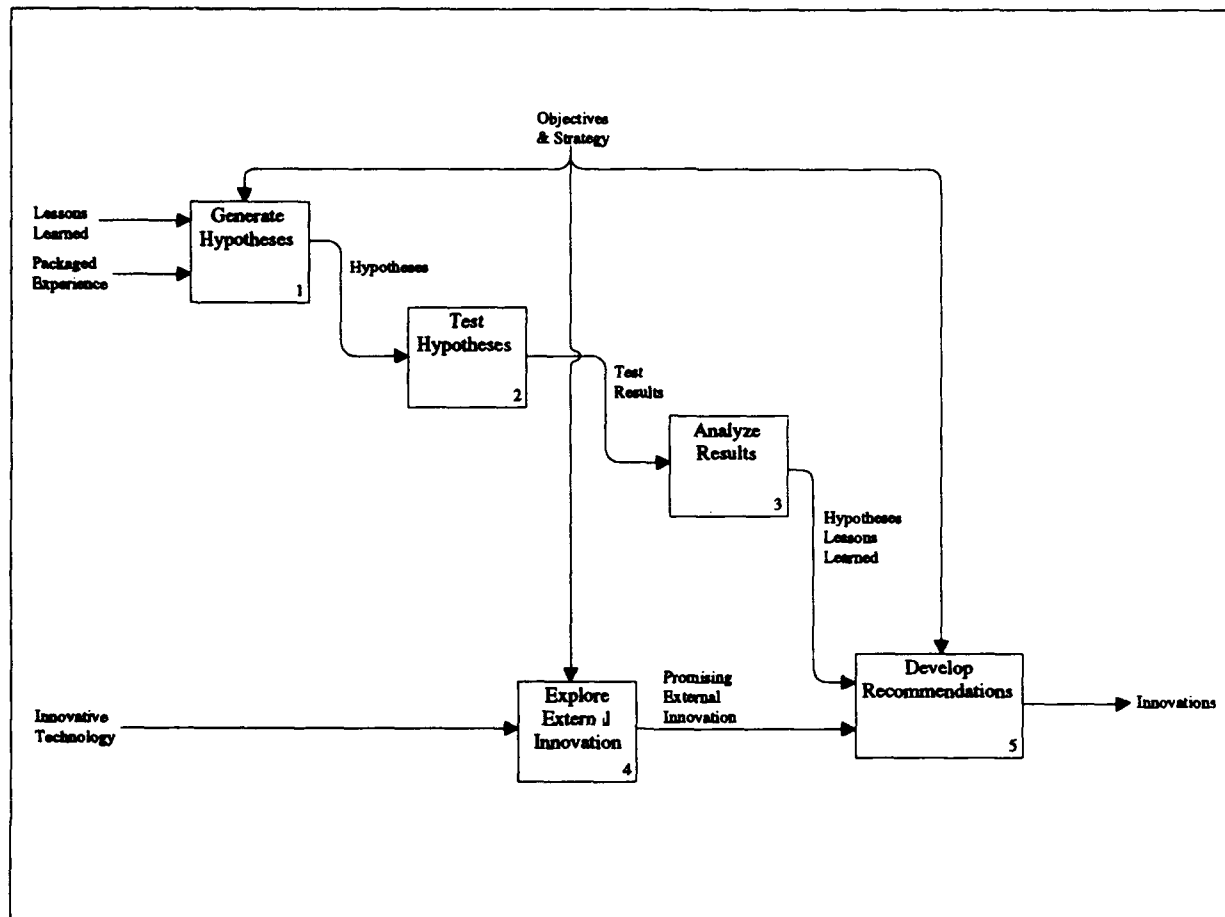
these process observations will inevitably need to be performed in tandem with reuse project activity, rather than taking place after completion of the project. The Observe Projects IDEF₀ diagram is shown in Figure 26. Observe Projects includes Conduct Observation, Solicit Information, Gather Metrics Data, and Package Experience.

The **Conduct Observation** process includes reflective observation by project team members, automated observation via instrumented tools and environments, and observations by outside researchers.

The **Solicit Information** process gathers information from project team members through interviews, questionnaires, debriefings, etc.

The **Gather Metrics Data** process gathers together the measurements obtained in the Monitor Project process.

The **Package Experience** process packages the observations, solicited information, and metrics data into a form that will be easy to analyze. This may include generalizing and formalizing the data and documenting project history data.

Figure 28: Explore Innovation IDEF₀ Diagram

3.3.3.2 Evaluate Projects

Evaluate Projects processes compare program objectives with the actual experience and data generated in Enactment and gathered in the Observe Projects process category. The intent of this evaluation is to assist in generating recommendations for incremental improvements within subsequent program cycles, or other programs, as opposed to recommendations to be applied immediately to the current projects being enacted. Even though these evaluation processes can conceptually be thought of as following the processes being evaluated, in practice the activities may often be concurrent, and may be performed by the same people.

The Evaluate Projects IDEF₀ diagram is shown in Figure 27. Evaluate Projects includes Model Experience, Analyze Metrics Data, Compare to Plans and Objectives, Learn from Infrastructure Usage, Identify Reusable Processes, and Review Results.

3.3.3.3 Explore Innovation

Explore Innovation processes gather, generate, analyze, and test new ideas, discoveries, and

innovations to generate recommendations for potentially dramatic improvements.

The Explore Innovation IDEF₀ diagram is shown in Figure 28. Explore Innovation includes Generate Hypotheses, Test Hypotheses, Analyze Results, Explore External Innovation, and Develop Recommendations.

The **Generate Hypotheses** process reviews Lessons Learned, Packaged Experience, and other data to develop hypotheses about possible improvements that can be made.

The **Test Hypotheses** process can occur using two different methods. The first method involves collecting empirical data from current projects to support conjectures based on the engineering results. The second method is to perform experimental projects to generate the empirical data to support conjectures based on the engineering results

The **Analyze Results** process analyzes the Test Results to develop Hypotheses Lessons Learned.

The **Explore External Innovation** processes gather potential assets, domain knowledge, and technology innovations from external sources

The **Develop Recommendations** process produce recommendations based on the Hypotheses Lessons Learned and Promising External Innovations. These recommendations can include changes to tools, environments, and processes; restructuring of organizational policies and procedures; changes in emphasis in organizational training materials; and nomination of ripe new target domains.

3.3.3.4 Recommend Enhancements

The Recommend Enhancements processes collect and filter the verified results of Explore Innovation and Evaluate Projects to the Organization Planning family of processes. The enhancements suggested by Innovation Exploration and Process Evaluation may be initially rejected as insufficient or inappropriate, thus initiating a further round of investigation.

The Recommend Enhancements IDEF₀ diagram is shown in Figure 29. Recommend Enhancements includes Synthesize Results, Analyze Feasibility, Perform Appraisals and Trade Offs, Determine Recommendations, and Analyze External Needs.

The **Synthesize Results** process synthesizes the results of a number of separate investigative activities into concrete recommendations.

The **Analyze Feasibility** process considers the *technical* feasibility or desirability of new approaches.

The **Perform Appraisals and Trade Offs** process appraises and trades off potential enhancements and innovations.

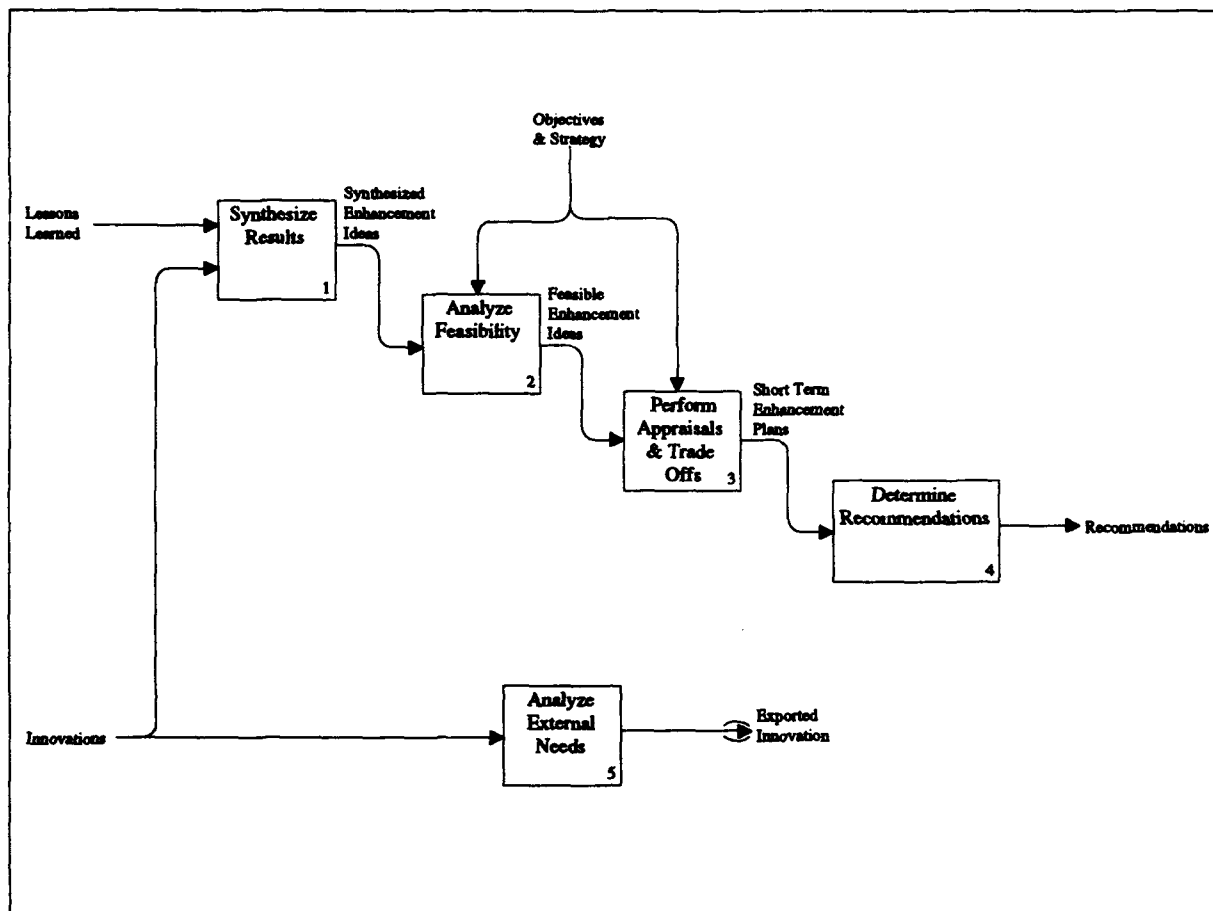


Figure 29: Recommend Enhancements IDEF₀ Diagram

The **Determine Recommendations** process makes recommendations based on the appraisals and trade offs.

The **Analyze External Needs** process determines which internal Innovations can be propagated outside the program to other programs and organizations or to the research community.

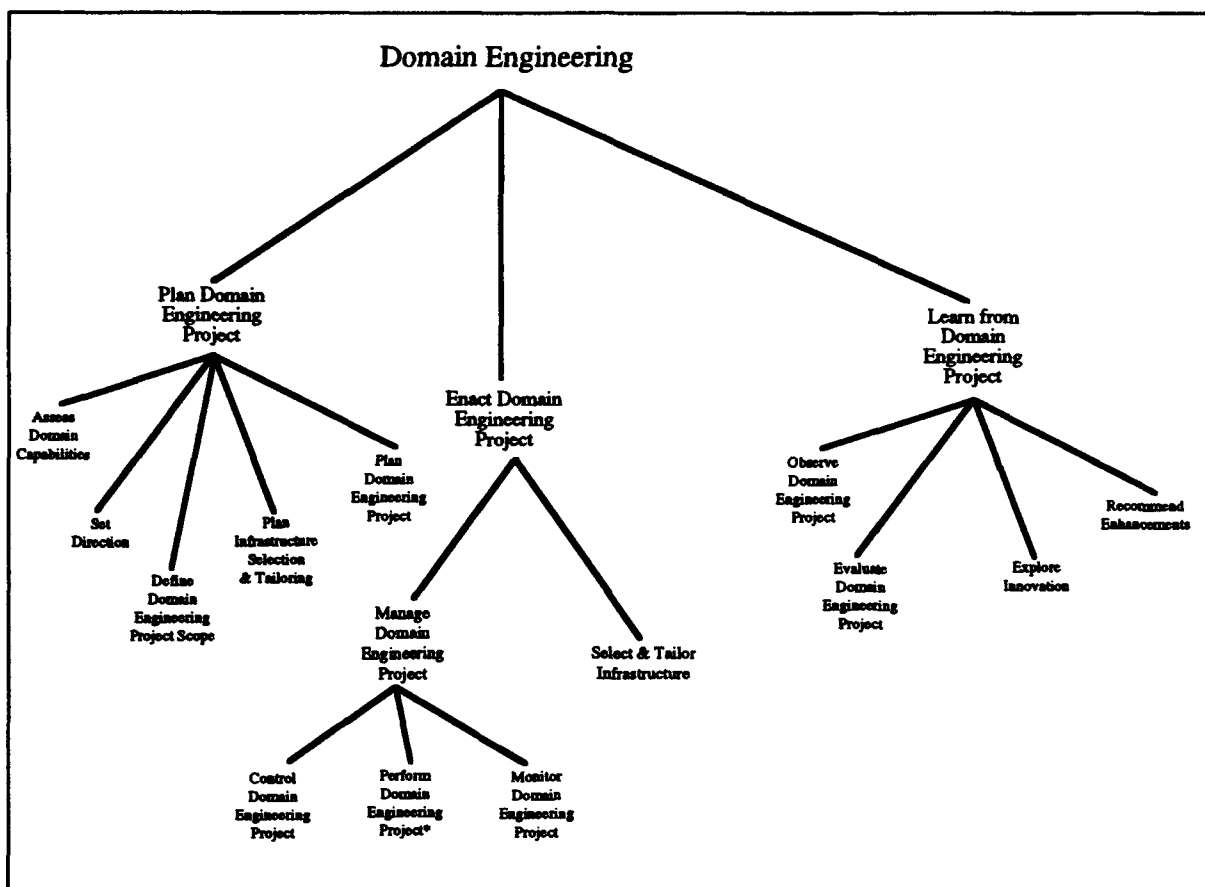


Figure 30: Domain Engineering Process Abstraction Hierarchy

3.4 Domain Engineering

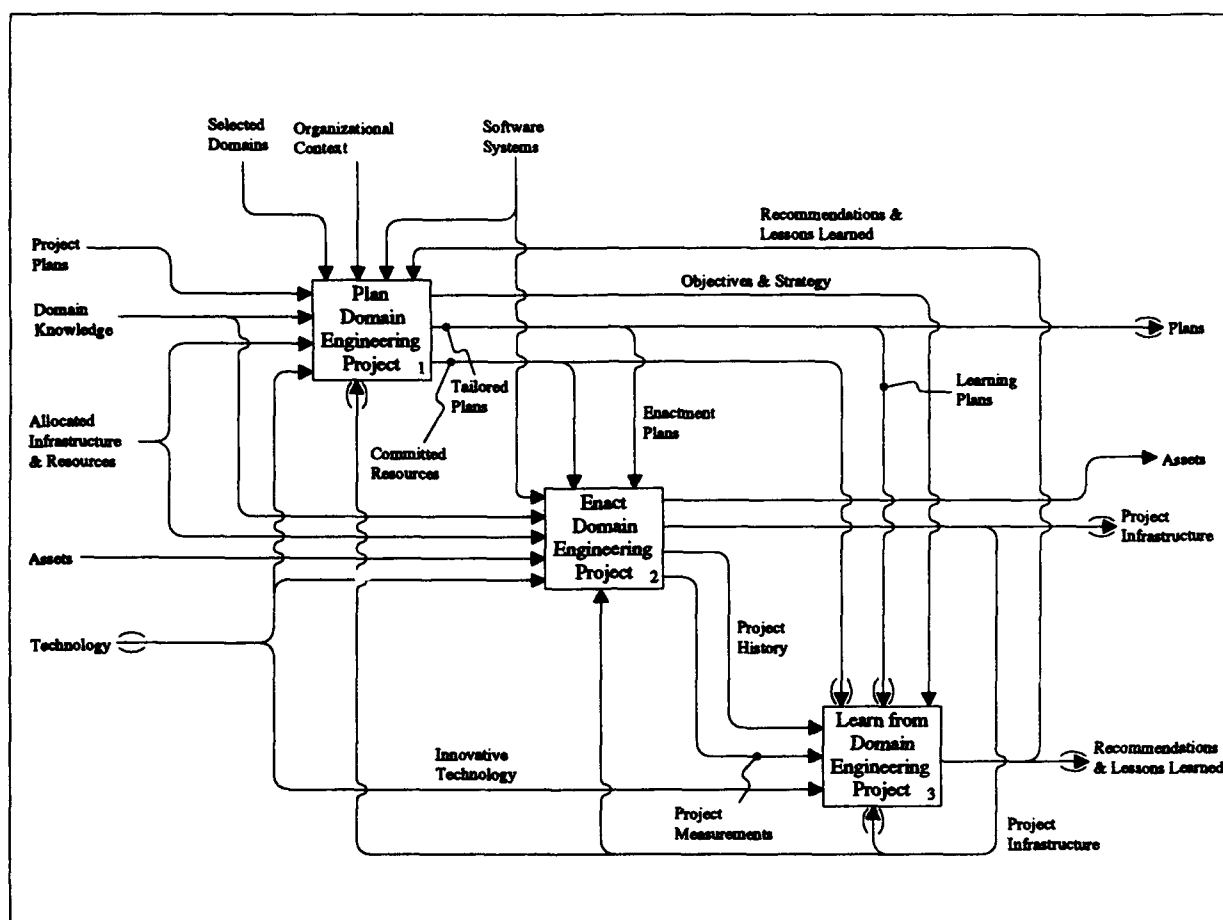
The goal of the ROSE Domain Engineering project submodel is to capture, organize and represent knowledge about a domain and produce reusable assets that can be applied to produce a family of systems encompassing that domain. Domain Engineering processes produce and evolve domain assets, including domain models and architectures and application generators as well as software components. The process abstraction hierarchy diagram for Domain Engineering is shown in Figure 30.

Figure 31 shows a table of the activities within Domain Engineering. The activities are broken down into Plan, Enact, and Learn process families. Project Performance is part of Enact, so the Project Performance process categories are embedded in the Enact process family in the table. The Domain Engineering Project Performance process categories include Analyze and Model Domain, Develop Software Architecture, Develop Application Generators, and Develop Software Components.

Another view of the Domain Engineering activities, showing the flow of information among them, can be seen in the IDEF₀ diagrams. Figure 32 shows the top-level IDEF₀ Diagram

Category \	Plan	Enact	Learn
Plan	Review Learning Insights Assess Domain Capabilities Set Direction Objectives & Strategy	Define Domain Eng. Project Scope Plan Infrastructure Selection & Tailoring	Plan Domain Eng. Project Identify Constraints Identify Domain Risks
Enact	Manage Domain Eng. Project Control Domain Eng. Project Perform Domain Eng. Project Monitor Domain Eng. Project	Select & Tailor Infrastructure	
1 Analyze and Model Domain	Plan Domain Analysis Forecast Requirements Determine Domain Reuse Goals Plan Domain Analysis Activities	Acquire Domain Knowledge Reverse Engineer Interview Domain Experts Review Domain Documentation Analyze Future Requirements Model Domain Define Domain Scope Domain Develop Domain Model Develop Domain Exemplar Set Develop Domain Genealogy Develop Domain Interconnection Model Create Domain Boundary Decision Rpt. Develop Feature Set Develop Domain Language Develop Domain Vocabulary Develop Domain Taxonomy	Validate Domain Model Perform Model Walkthroughs Review Domain Model Conduct Expert Reviews Validate Domain Model Analyze Results
2 Develop Software Architecture	Plan Architecture Development Determine Domain Architecture Objectives Perform Constraint Control Assess Domain Risks Develop Domain Risk Mitigation Plan Develop Architecture Test & Validation Plan Select Technology Develop Phased Implementation Plan	Acquire Architecture Information Reverse Engineer Perform Design Recovery Develop Domain Architecture Define Domain Architecture Create Architecture Prototypes Determine Architecture Interfaces Determine Architecture Usage Principles Determine Arch. Mapping to Reqmts Create Architecture Variations Prioritize Variations Feature Variant Prioritization Separable Selectability Trade Off Domain Implementation Model Incl. Component Relationships	Validate Domain Architecture Perform Architecture Walkthroughs Conduct Expert Reviews
3 Develop Application Generators	Plan Applic. Gen. Development Generator Implementation Constraints Gen. Acceptance Constraints Risk Plan Refinement Gen. Development Guidelines Develop Applic. Gen. Test Plan	Develop Generators Analyze Variations Develop & Evolve Generic Generators Define Generator Usage Principles Create Documentation Map Application Generators to Architecture	Validate Applic. Generators Perform Generator Walkthroughs Conduct Expert Reviews Test Generators Validate Generators to Domain Model & Architecture
4 Develop Software Components	Plan Component Development Component Implementation Constraints Comp. Acceptance Constraints Risk Plan Refinement Comp. Development Guidelines Develop Component Test Plan	Develop Components Develop & Evolve Generic Components Define Component Usage Principles Create Documentation Tailor Process Assets to Domain Map Components to Architecture	Validate Components Perform Component Walkthroughs Conduct Expert Reviews Test Components Validate Components to Domain Model & Architecture
Learn	Observe Domain Eng. Project	Evaluate Domain Eng. Project	Explore Innovation Recommend Enhancements

Figure 31: Domain Engineering Table

Figure 32: Domain Engineering IDEF₀ Diagram

for Domain Engineering. This diagram shows that the **Domain Engineering Process** is broken hierarchically into the **Plan Domain Engineering Project**, **Enact Domain Engineering Project**, and **Learn from Domain Engineering Project** Processes.

Domain Engineering inputs include external **Assets** that were created by Domain Engineering processes within a different organization; **Domain Knowledge** that can be imparted in a variety of ways to provide information about the domain; **Project Plans** inherited from organizational program planning; **Allocated Infrastructure and Resources** that can be tailored to the project; and **Technology** that can contribute to the project's infrastructure and can be applied to automate processes. Controls on Domain Engineering include existing **Software Systems** that can provide legacy information about a domain and potentially provide a source of raw material for assets; **Selected Domains** that are the focus of the Domain Engineering project; and **Organizational Context** that guides and constrains various aspects of the project to ensure that they are consistent with overall organization strategies, policies, etc. Outputs from Domain Engineering include project **Plans**; the tailored **Project Infrastructure**; **Recommendations and Lessons Learned** from project management; and **Assets** in the domain including Domain Information Models, Application Generators,

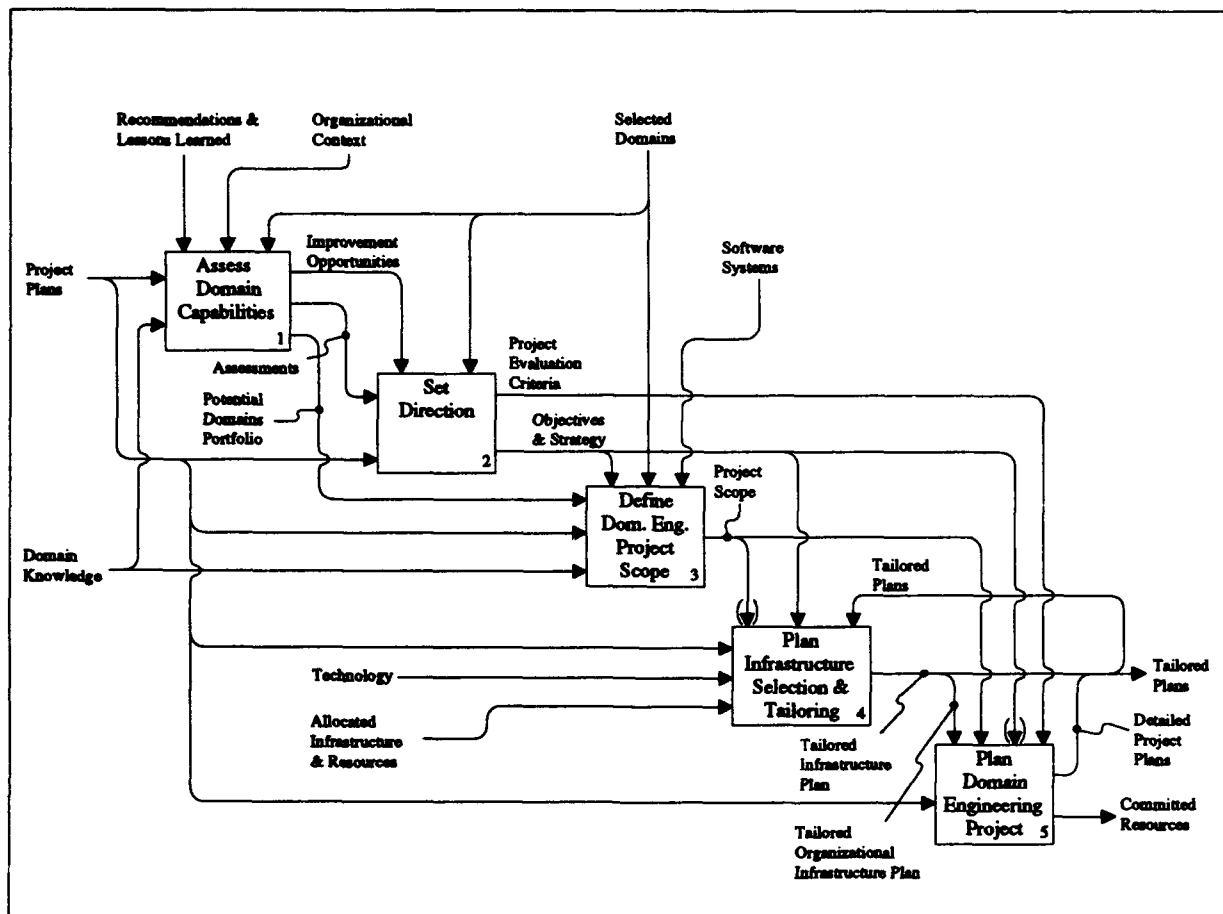


Figure 33: Plan Domain Engineering Project IDEF₀ Diagram

and Software Components. Domain Information Models describe generic domain requirements and solutions, including Domain Requirements Models, Domain Architecture Models, and Domain Implementation Models.

The following subsections describe the activities carried out in a Domain Engineering Project. These are divided into the Plan-Enact-Learn project management activities described briefly in Section 3.4.1, and the Asset Creation engineering activities (performed within the Enact family) described in Section 3.4.2.

3.4.1 Manage Domain Engineering

This section describes the processes carried out in managing a Domain Engineering project. The processes are described briefly since they are generally the same set of processes described in Organization Management, except with a project focus. The differences between Domain Engineering project management and Organization Management are stressed in this section.

3.4.1.1 Plan Domain Engineering Project

Figure 33 contains an IDEF₀ Diagram for the Plan Domain Engineering Project process family. As shown in the figure, the Plan process consists of the following five process categories:

- **Assess Domain Capabilities** processes characterize the current state of Domain Engineering practice and capabilities within the organization. The assessment considers a variety of factors, including domain engineering experience, domain expertise, available support technology, and availability of legacy systems and pre-existing assets. Included in this process is a review of recommendations and lessons learned from previous cycles to identify specific improvements that can be made to the organization's Domain Engineering capabilities.
- **Set Direction** processes determine a strategy and objectives for the Domain Engineering project. A high-level strategy may be defined in the Plan Projects process category within Organization Management. In Set Direction, a more detailed strategy and objectives are defined, based on the high-level strategy. Once a strategy and objectives have been developed, success criteria are identified to determine whether the objectives are met.
- **Define Domain Engineering Project Scope** processes bound the scope of key aspects of the project. This may include defining the boundaries of the domain in more detail than was done in the Scope Line of Business process in Organization Management, although this may be left in whole or in part to the Analyze and Model Domain processes. This process may also establish guidelines for the breadth and depth of the domain analysis, impose constraints on the quality or adaptability of the domain assets created, and so on.
- **Plan Infrastructure Selection and Tailoring** processes plan the technical, organizational, and educational infrastructure needs of the project. These needs can be satisfied by the infrastructure at the organizational level, tailored from the infrastructure at the organizational level, or, if necessary, created specifically for the project. Project infrastructure planning includes plans for processes, policies, standards, and training.
- **The Plan Domain Engineering Project** processes perform detailed project planning. Detailed project planning includes the development of a schedule, budget, and resource needs based on the high-level schedule and budget developed in the Plan Projects process category in Organization Management. Political, technical, schedule, and cost constraints on the project are identified and control mechanisms are planned. Domain risks are identified and evaluated as "high", "medium", or "low". Risk mitigation plans and mechanisms are developed for handling high risks.

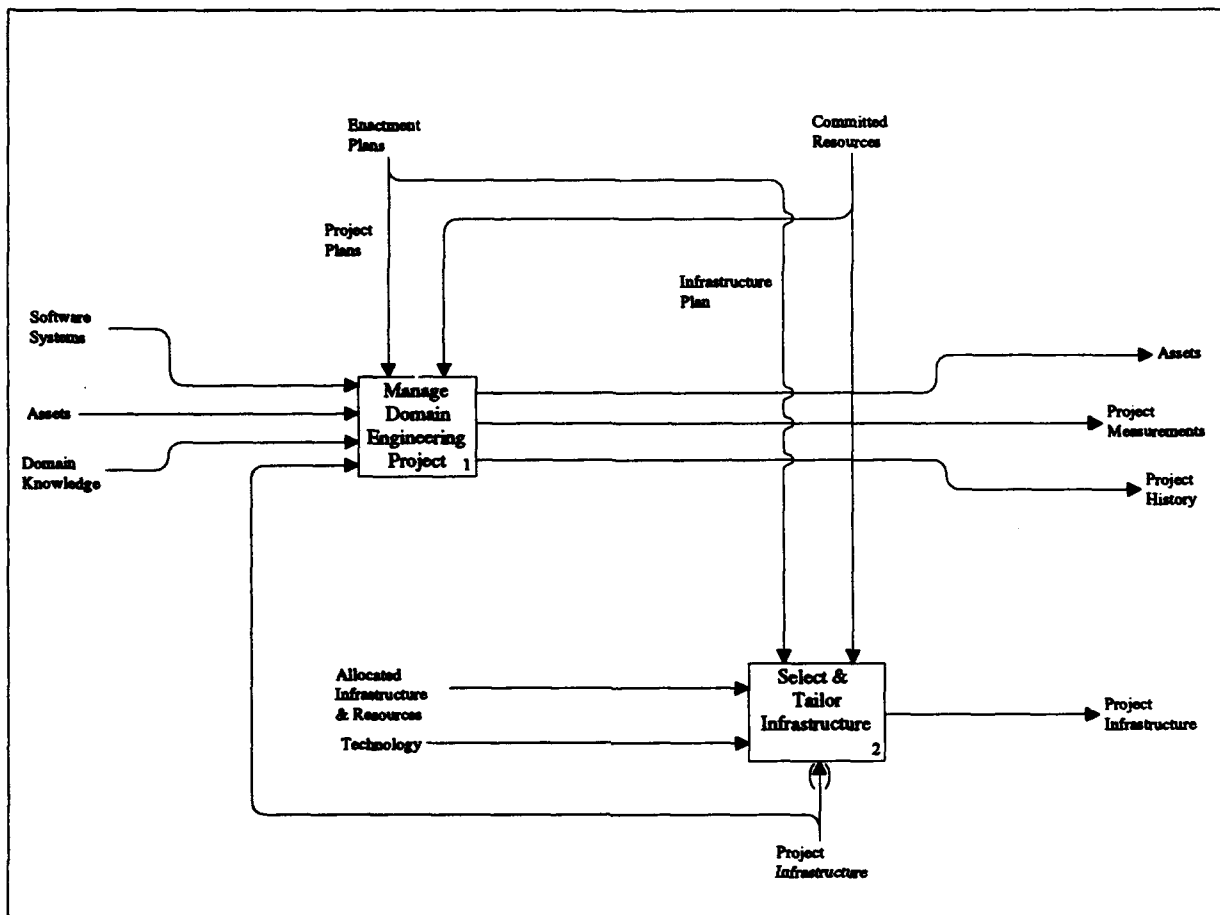


Figure 34: Enact Domain Engineering Project IDEF₀ Diagram

3.4.1.2 Enact Domain Engineering Project

The Enact Domain Engineering Project process family manages the day-to-day activity of the project and ensures that an infrastructure sufficient to meet the needs of the project is established and maintained.

Figure 34 contains an IDEF₀ diagram for the Enact Domain Engineering Project process family. As shown in the figure, the Enact process family consists of the **Manage Domain Engineering Project** and **Select and Tailor Infrastructure** process categories.

Manage Domain Engineering Project

The Manage Domain Engineering Project process category establishes a temporal context for the performance of the Domain Engineering project and performs detailed project supervision. The Manage Domain Engineering Project IDEF₀ diagram is shown in Figure 35. The Manage Domain Engineering Project process includes Control Domain Engineering Project, Perform Domain Engineering Project, and Monitor Domain Engineering Project.

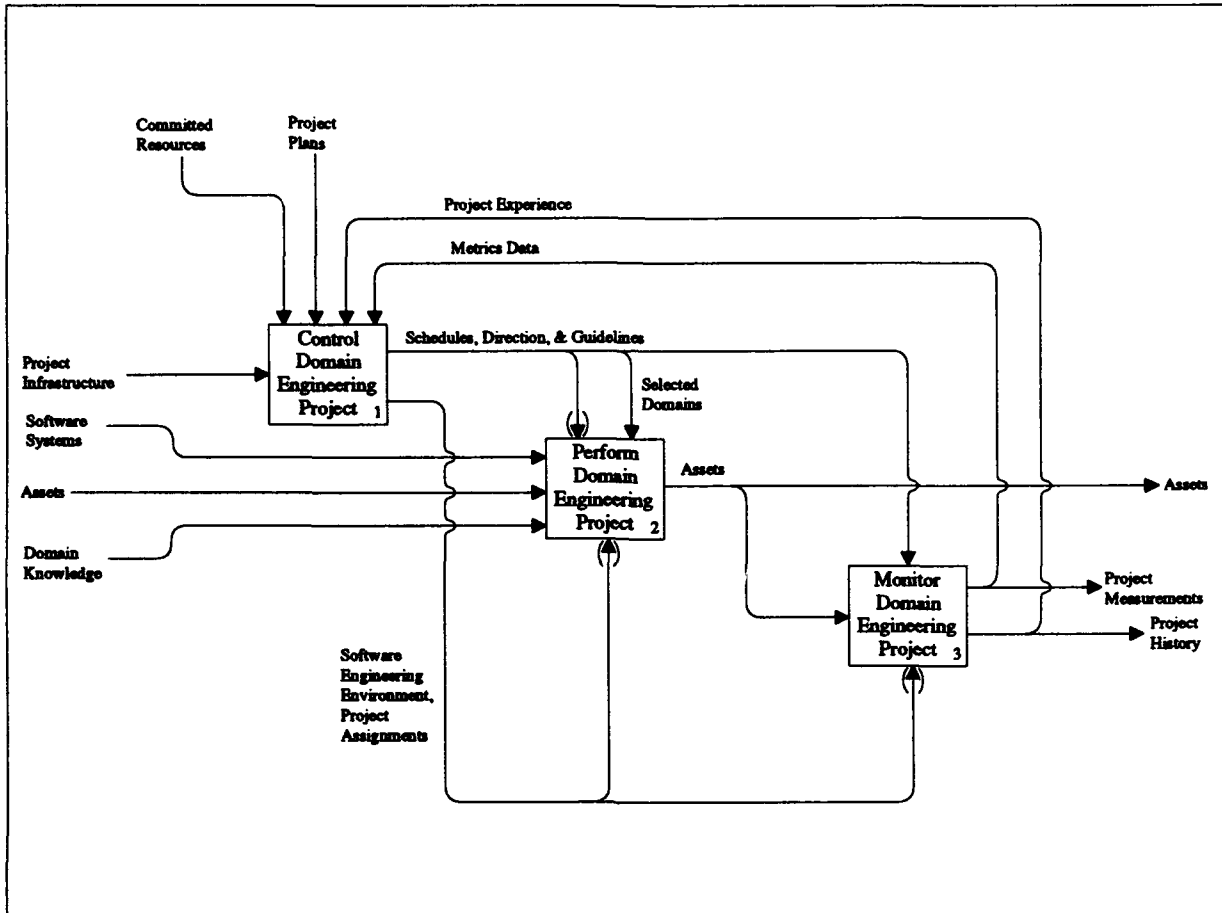


Figure 35: Manage Domain Engineering Project IDEF₀ Diagram

The **Control Domain Engineering Project** process activities intervene with project performance in order to keep the project on track relative to objectives set during Organizational Planning. Project control is the “management” function (in the most conventional sense) for the Domain Engineering project. The goal of project control activities is to optimize overall project performance, as measured by factors such as the rate of growth of the asset population, the degree to which the assets are utilized, the overall quality of the assets and resulting systems, the effectiveness of the infrastructure, etc.

The **Perform Domain Engineering Project** activities at the core of Enactment are where the engineering processes being enacted are “hooked in” to Domain Engineering project management and actually performed by individual staff members. These engineering processes are described in Section 3.4.2. From a management perspective, project performance activities also involve tactical decisions about *how* the work will be performed on a detailed, day-to-day basis. At one level, project performance activities can be viewed as individualized techniques for filling in the minute implementation details that are generally missing from project processes established in Project Planning.

The **Monitor Domain Engineering Project** activities capture “learning-oriented” in-

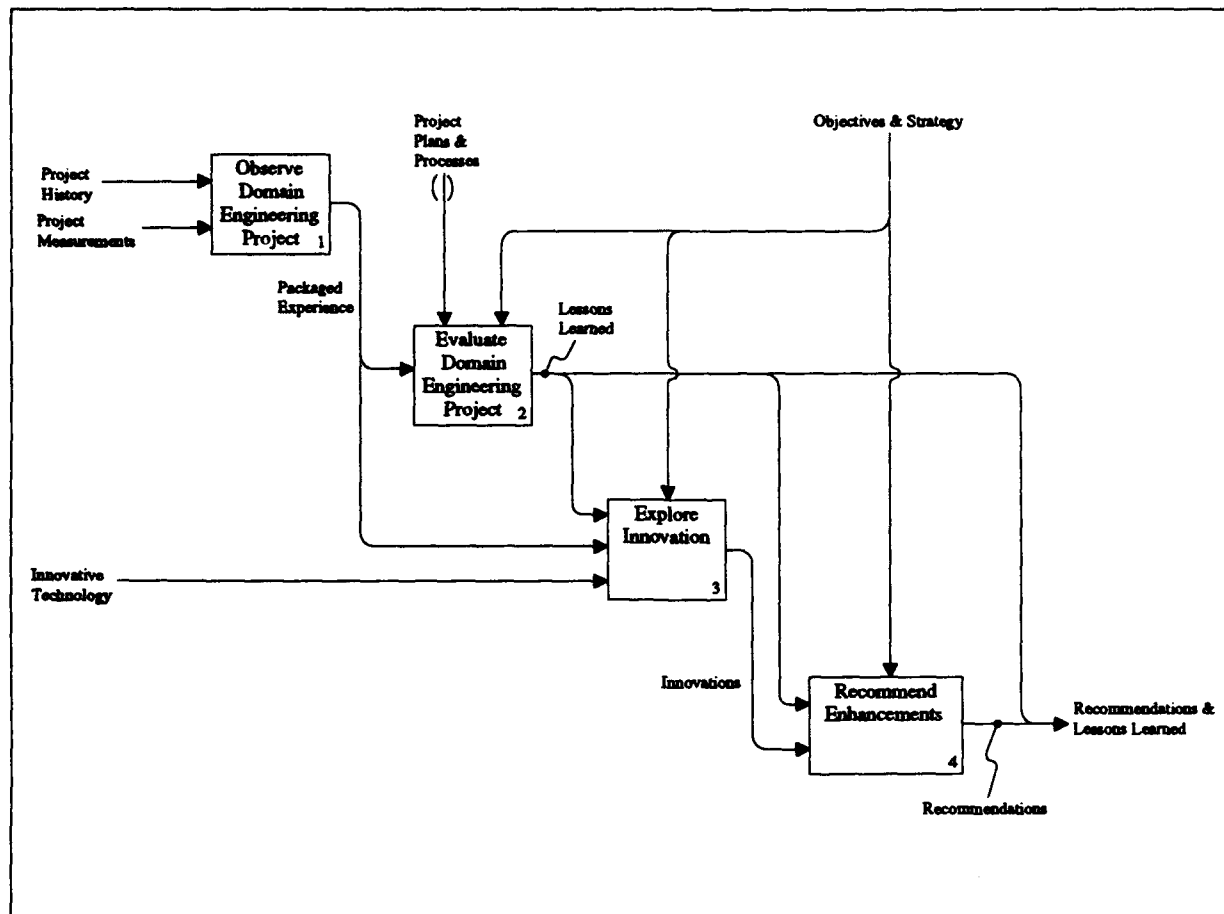


Figure 36: Learn from Domain Engineering Project IDEF₀ Diagram

formation from the Domain Engineering project as it is performed. Some of this information provides feedback to project control activities in order to adjust schedules, budgets, milestones, incentives, guidelines, or task assignments. Some of the same information may serve as input to the Learn from Domain Engineering Project processes, along with data collected in accordance with process and product metrics identified in Plan Domain Engineering Project, and project history data such as rationale for decisions made and interim work products created.

Select and Tailor Infrastructure

Select and Tailor Infrastructure processes select the infrastructure to meet the project's needs from the organization's infrastructure. This infrastructure may be tailored to the project and missing infrastructure capabilities to satisfy new project-specific needs may be developed.

3.4.1.3 Learn from Domain Engineering Project

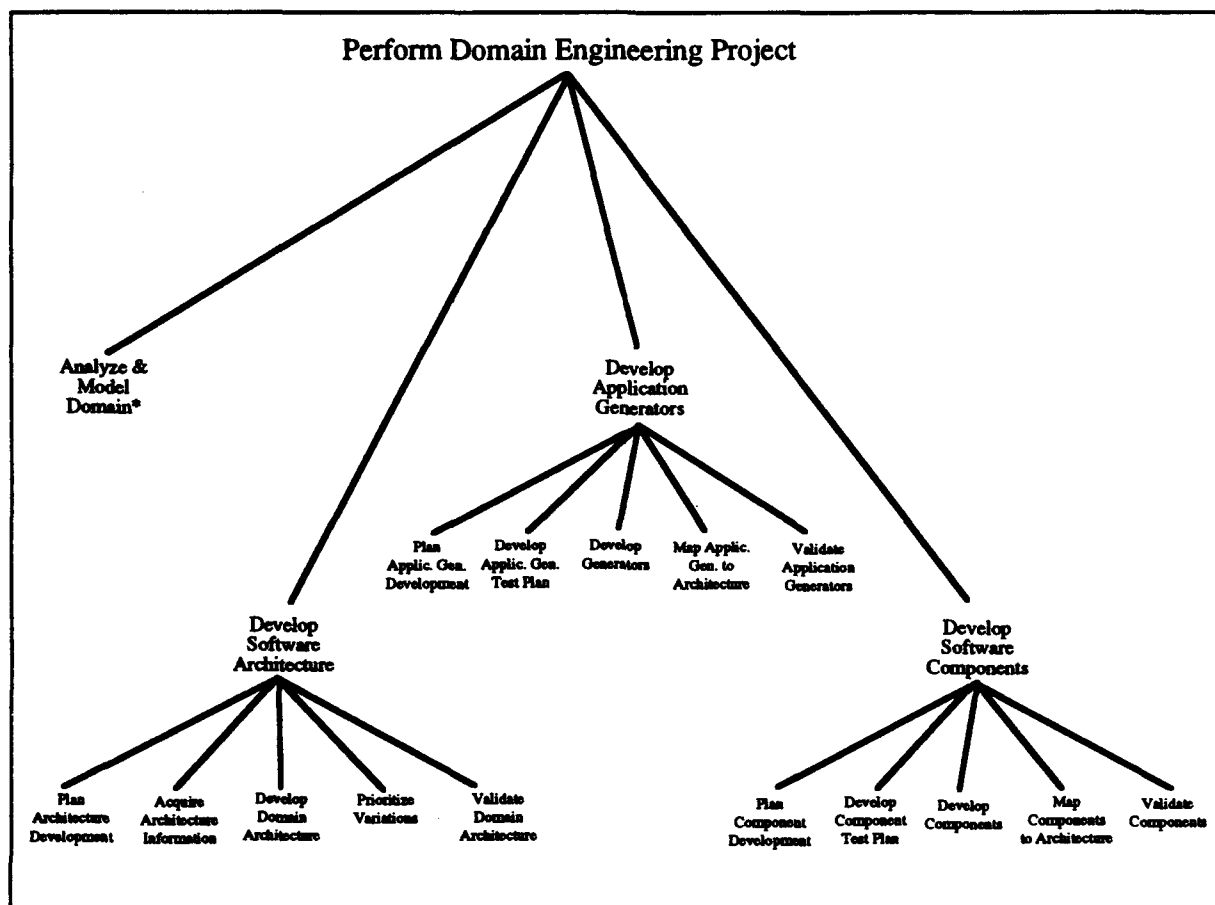


Figure 37: Perform Domain Engineering Project Process Abstraction Hierarchy

Processes in the Learn from Domain Engineering Project process family assess the overall success of the strategies and plans put into place by the Plan Domain Engineering Project process family, comparing project results and experience against project objectives, to improve the quality of the plans and infrastructure. Figure 36 contains an IDEF₀ diagram for the Learn from Domain Engineering Project process family. As shown in the figure, the Learn process family consists of **Observe Domain Engineering Project**, **Evaluate Domain Engineering Project**, **Explore Innovation**, and **Recommend Enhancements**. These process categories were discussed in section 3.3.3, on Organization Management Learning. Project level learning is similar to organizational learning, except that the focus is on the individual Domain Engineering project.

3.4.2 Perform Domain Engineering

The **Perform Domain Engineering Project** process included among the Enact Domain Engineering Project processes described above is where the engineering activities enacted in the project are "hooked in" to project management activities. This section describes these

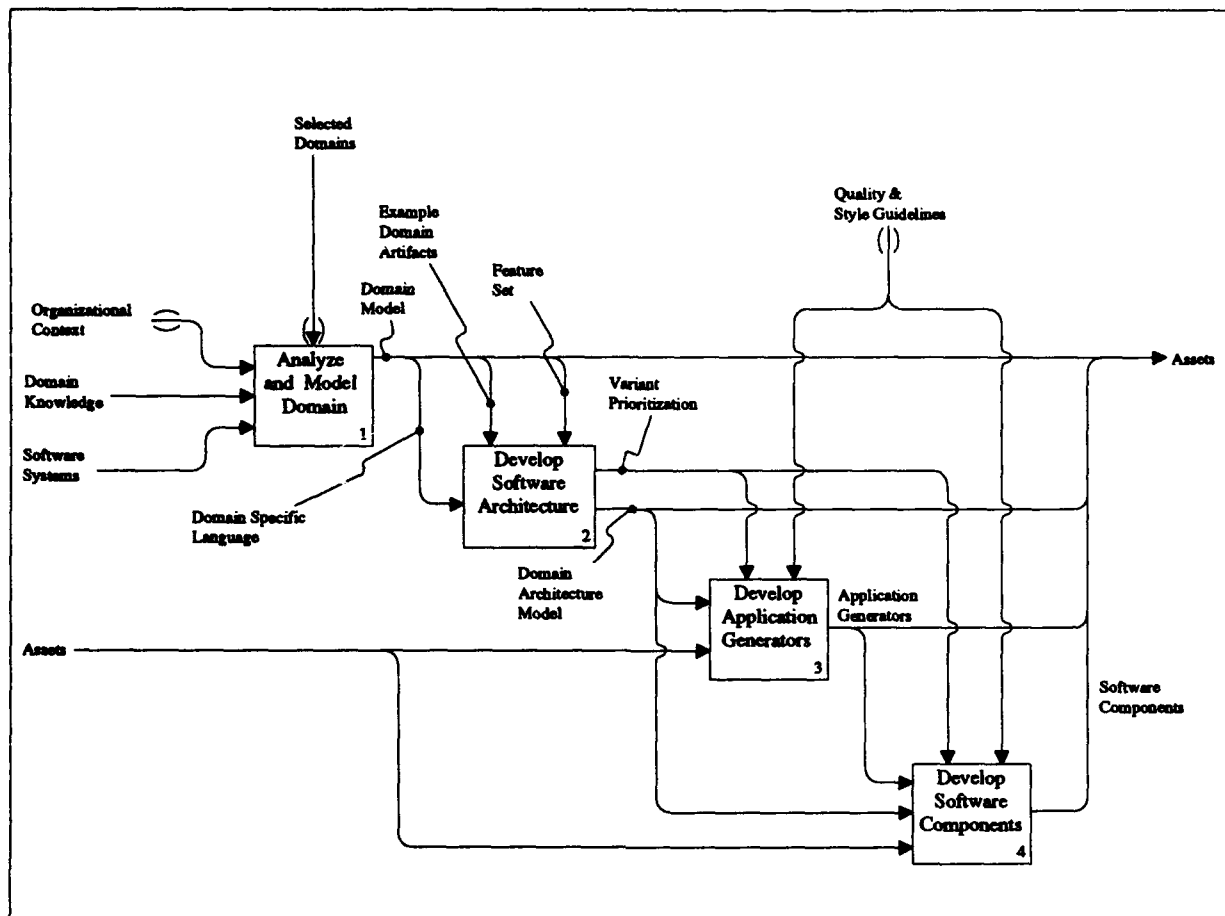


Figure 38: Perform Domain Engineering Project IDEF₀ Diagram

engineering activities, divided into process categories that include their own localized Plan-Enact-Learn activities. The process abstraction hierarchy diagram for Perform Domain Engineering Project is shown in Figure 37.

The Perform Domain Engineering Project IDEF₀ diagram is shown in Figure 38. Perform Domain Engineering Project includes Analyze and Model Domain, Develop Software Architecture, Develop Application Generators, and Develop Software Components. Each of these process categories is described in detail below.

3.4.2.1 Analyze and Model Domain

Analyze and Model Domain processes focuses on producing engineering models that characterize domain requirements. Other activities that are sometimes associated with domain analysis, but which focus more on organizational and management-oriented issues, are addressed in the ROSE PM within Organization Management and Domain Engineering Management process categories such as Assessment and Domain Selection.

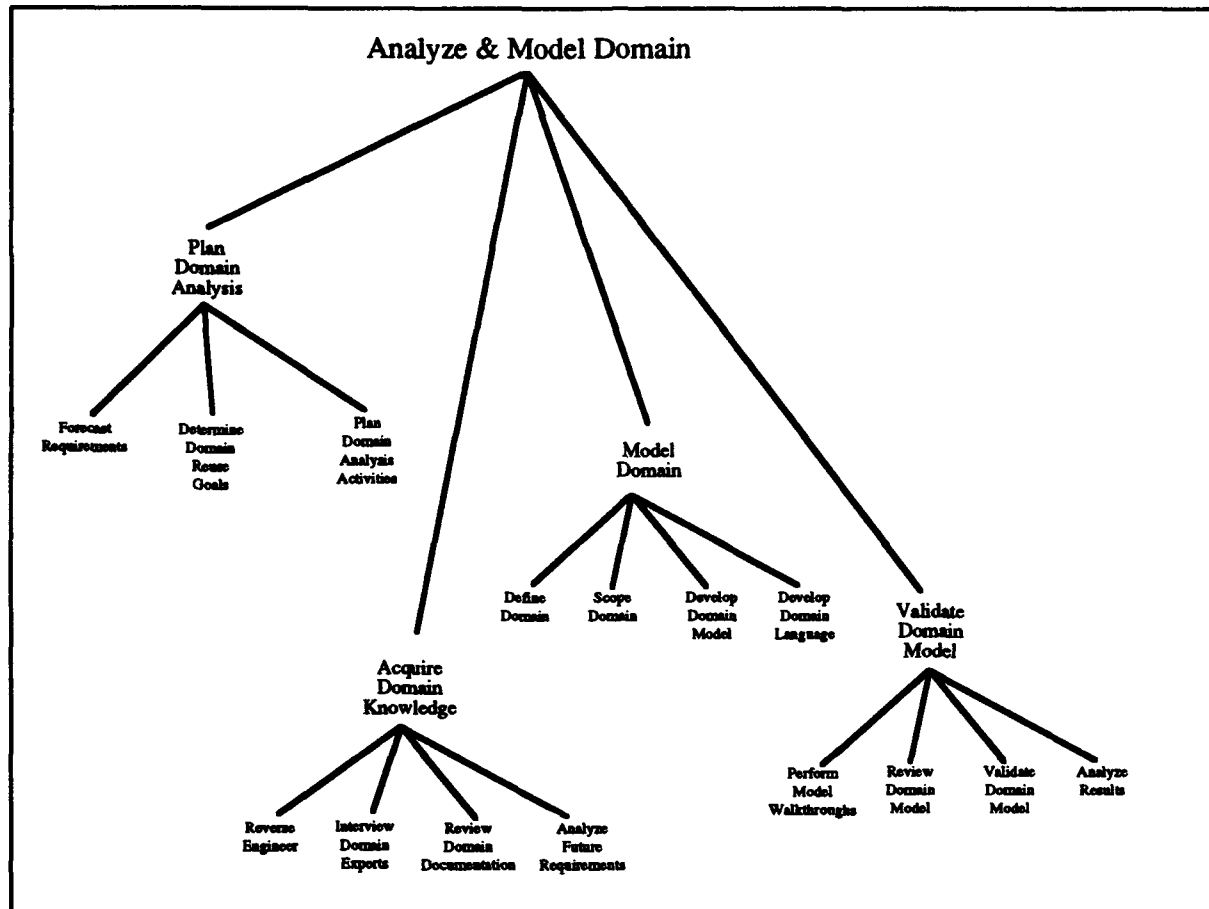
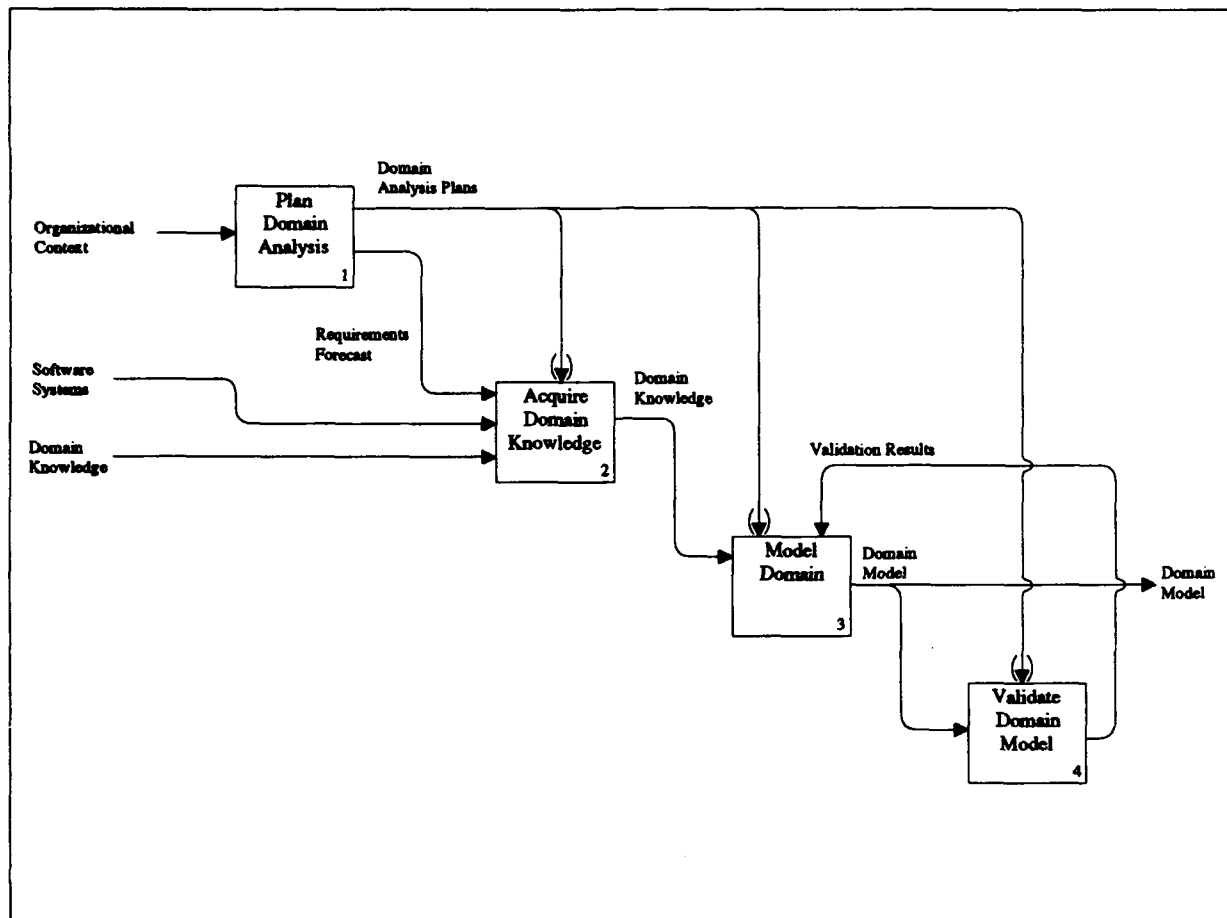


Figure 39: Analyze and Model Domain Process Abstraction Hierarchy

The principal goal of domain analysis is to develop a *domain model* (also called *domain requirements model* above) that characterizes the domain problem space in terms of a set of generic requirements on systems within the domain. In this context, the term “requirements” is used quite broadly to include the concepts and vocabulary on which the domain is based, the functional domain context, and the features and behavioral characteristics that are more commonly interpreted as requirements. There are also requirements on how these elements interrelate, and together all these requirements form a set of *domain constraints*. A central aspect of the domain model that differentiates it from a generic system model is that it describes how the domain constraints may vary among valid systems in the domain. It also provides rationale, rules of thumb, and other heuristics to assist in using the information in the model during Application Engineering. The process abstraction hierarchy diagram for Analyze and Model Domain is shown in Figure 39.

The Analyze and Model Domain IDEF₀ diagram is shown in Figure 40. Analyze and Model Domain includes the Plan Domain Analysis, Acquire Domain Knowledge, Model Domain, and Validate Domain Model processes.

Figure 40: Analyze and Model Domain IDEF₀ Diagram

Plan Domain Analysis

The Plan Domain Analysis IDEF₀ diagram is shown in Figure 41. Plan Domain Analysis includes the Forecast Requirements, Determine Domain Reuse Goals, and Plan Domain Analysis Activities processes. These processes may refine the objectives and the scoping or process decisions inherited from the overall project planning processes.

- **Forecast Requirements** processes forecast future requirements for the domain. In order for reuse to remain viable within typical domains over a period of years so that a return on the domain engineering investment will be fully realized, forecasting of future trends in domain requirements and supporting technology is essential. This forecasting activity can involve a wide variety of activities, including consulting domain and technology experts, keeping abreast of evolving standards and underlying technology, and staying directly involved in relevant domain and technology professional communities via conferences, journals, etc.
- **Determine Domain Reuse Goals** processes determine the goals for the domain models, domain architectures, domain application generators, and domain components

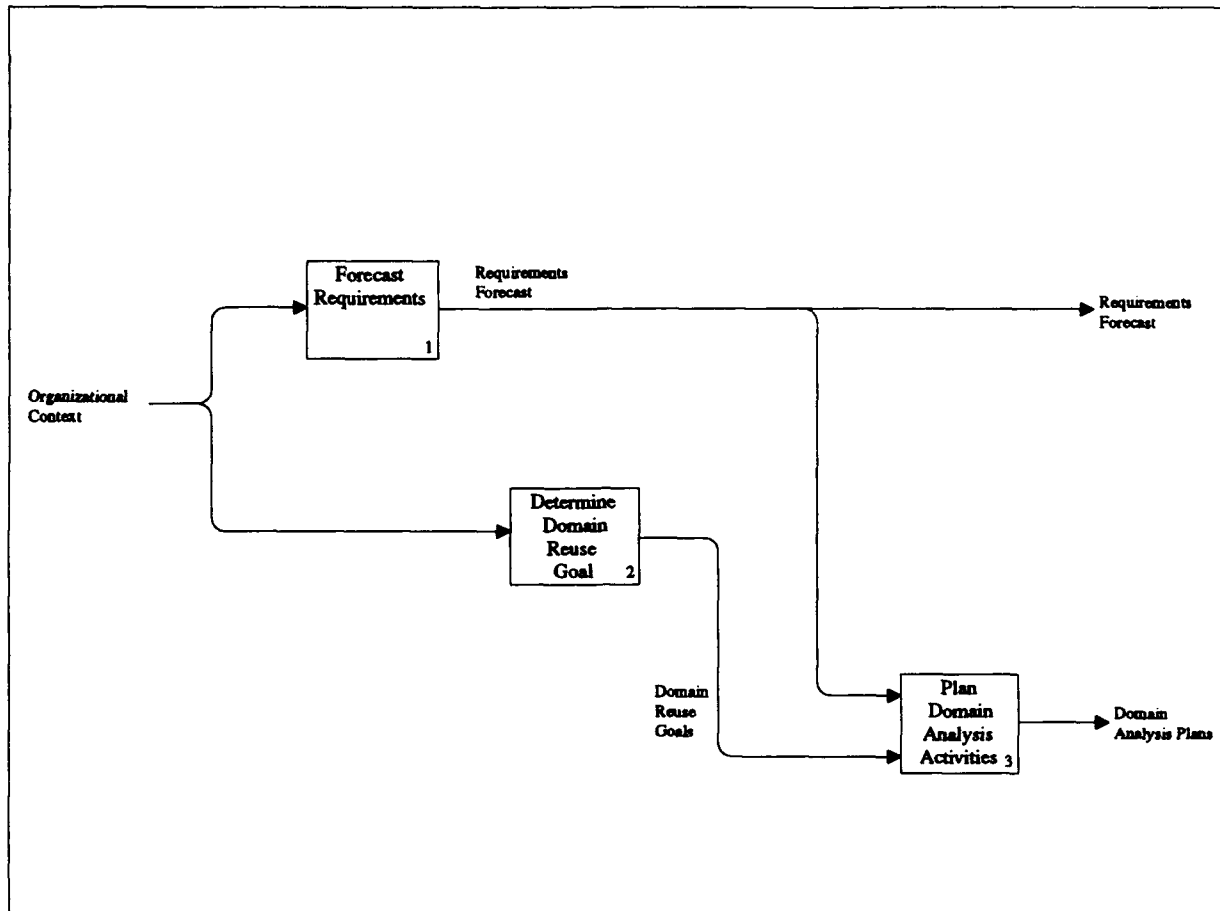


Figure 41: Plan Domain Analysis IDEF₀ Diagram

to be created, in terms of the needs of specific customers (or stakeholders) with vested interests in the products.

- **Plan Domain Analysis Activities** processes plan the specific steps to be used in carrying out domain analysis, including identification of the kinds of models that will be built and possibly the techniques that will be used to build them. The planned activities may vary from those described below, depending on the models and methods desired, and possibly other criteria. The detailed activities described in the ROSE PM suggest a particular approach, but alternative approaches can be readily applied within the broad ROSE framework.

Acquire Domain Knowledge

Acquire Domain Knowledge processes capture and formalize domain knowledge that would otherwise remain only in people's heads. Such knowledge is useful not only because it represents expertise and experience in the domain, but also because it can be used to verify or corroborate the information obtained through the analysis of existing systems.

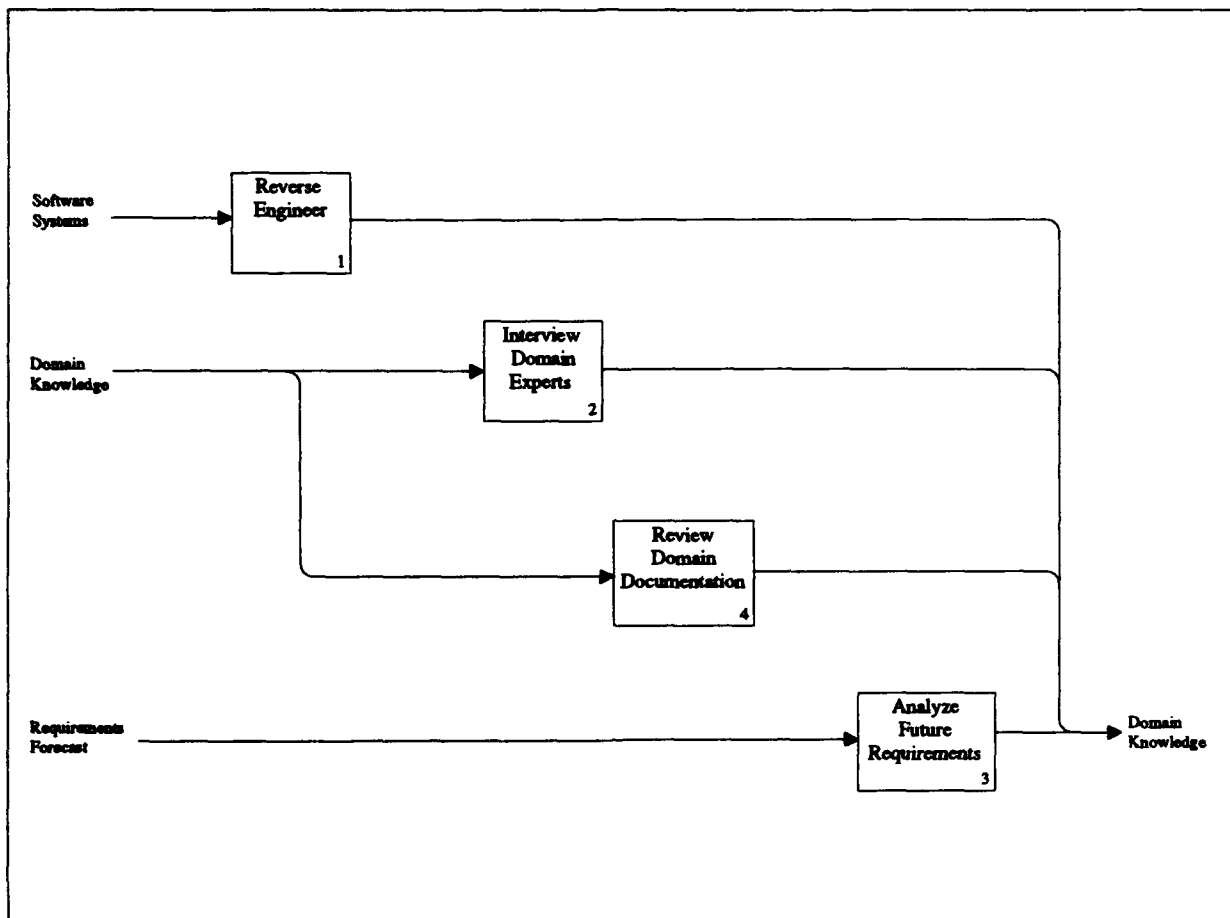
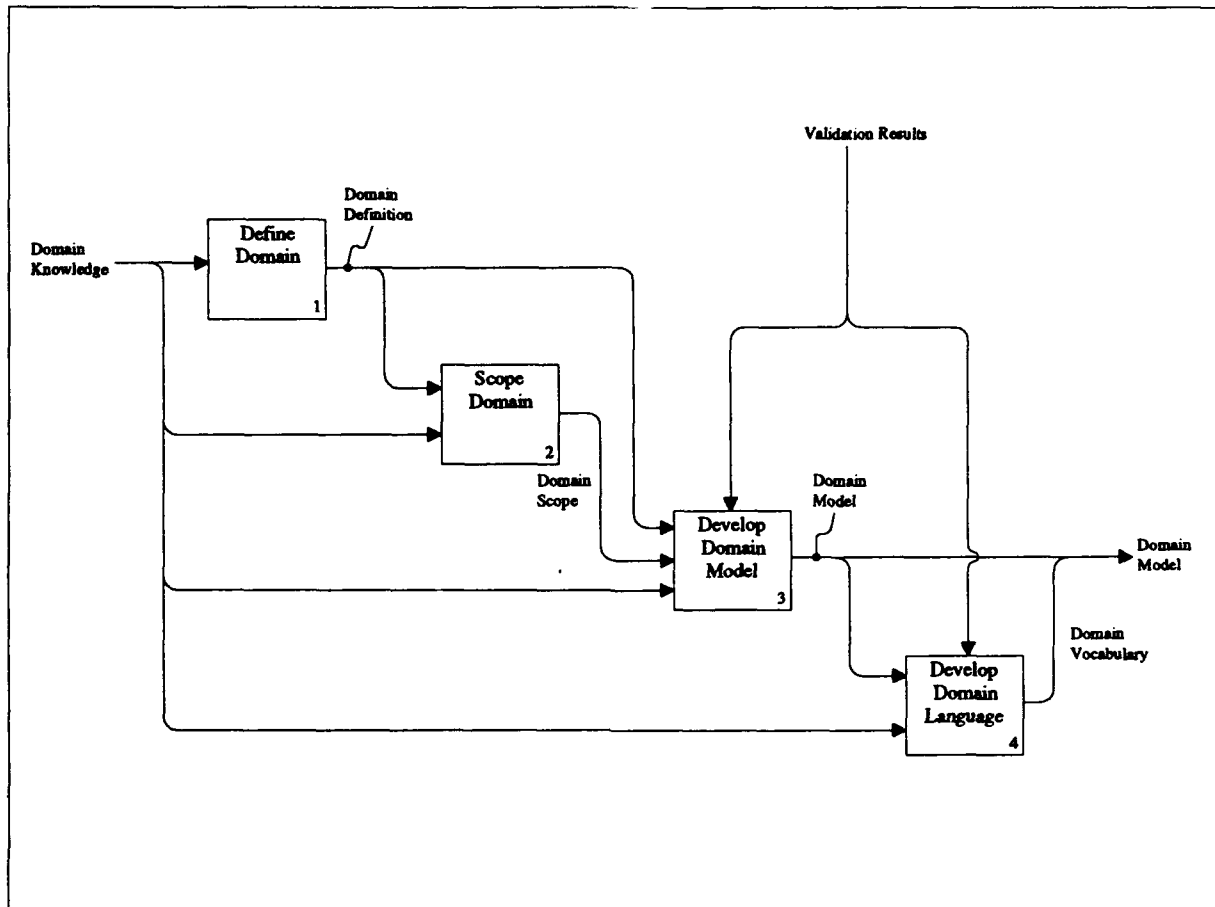


Figure 42: Acquire Domain Knowledge IDEF₀ Diagram

The Acquire Domain Knowledge IDEF₀ diagram is shown in Figure 42. Acquire Domain Knowledge includes the Reverse Engineer, Interview Domain Experts, Review Domain Documentation, and Analyze Future Requirements processes.

- **Reverse Engineer** processes are often used to extract expertise already encoded in legacy systems. Reverse engineering and design recovery techniques aid in identifying basic domain concepts and vocabulary, commonalities and patterns of variation in the domain, and, where several versions of the same system are available for analysis, responses to changes in technology.
- **Interview Domain Experts** processes support knowledge acquisition by adapting knowledge acquisition techniques used by expert system developers, interviewing techniques used for systems analysis and requirements elicitation, and general methods used for in-depth interviewing in any discipline.
- **Review Domain Documentation** processes review documentation from current systems within the domain in order to extract relevant domain information. This doc-

Figure 43: Model Domain IDEF₀ Diagram

umentation can include formal system documentation, informal documentation, published sources, electronic mail, and training materials.

- **Analyze Future Requirements** processes analyze and describe future requirements for systems within the domain to establish a baseline for such requirements as input to Model Domain processes. This process may involve critical analysis and interpretation of potential domain requirements identified both in the Forecast Requirements process and during other Acquire Domain Knowledge processes.

Model Domain

Model Domain processes characterize the domain in the form of a Domain Requirements Model. The Model Domain IDEF₀ diagram is shown in Figure 43. Model Domain includes the Define Domain, Scope Domain, Develop Domain Language, and Develop Domain Model processes.

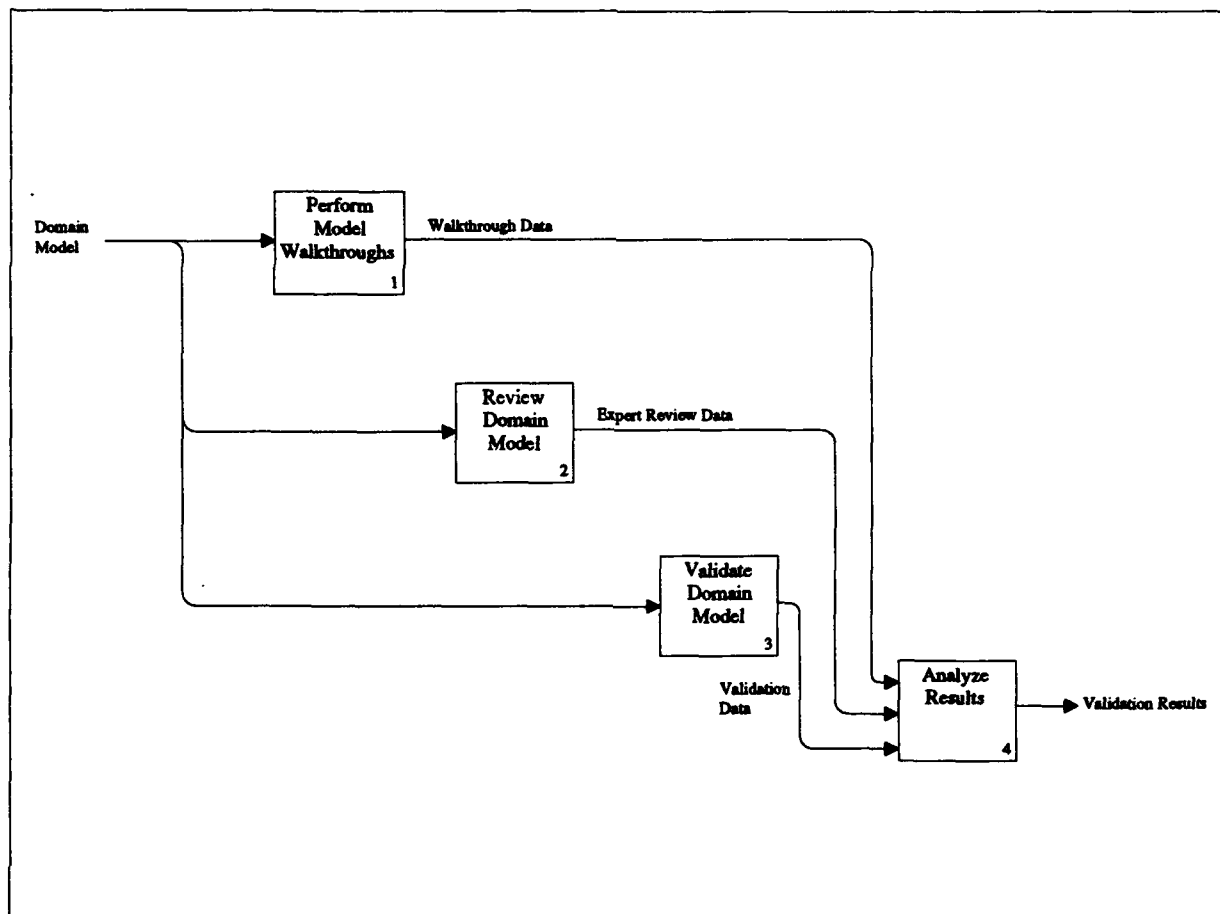
- **Define Domain** processes define the domain more formally than in the Scope Line of Business or Define Domain Engineering Project Scope activities performed at outer levels of planning. The primary purpose of domain definition is to clarify what systems or system functionality is included in the domain and what systems are excluded from the domain. Domain definition also involves identifying a set of legacy systems that exemplify the domain.
- **Scope Domain** processes determine the boundaries established around the domain. Relationships between the domain of focus and related domains are identified. A Domain Genealogy Model may be created to encapsulate data on the evolution and historical interrelationships and dependencies among systems within the domain.
- **Develop Domain Language** processes develop a domain vocabulary and develop or select a set of languages for describing aspects of the domain in terms of the vocabulary.
- **Develop Domain Model** processes create a domain requirements model. One approach to domain requirements modeling could be to develop descriptive domain models of current systems in the domain and then abstract them to a prescriptive model that prescribes the characteristics of domain assets (including domain architectures) that will be constructed to address future system needs. The prescriptive model should fully characterize the domain problem space in terms of the variability of requirements on assets in the domain.

Validate Domain Model

Validate Domain Model processes validate the domain requirements model. These processes should not necessarily be viewed as operating in isolation from other asset creation activities. In particular, the Develop Software Architecture, Develop Application Generators, and Develop Software Components processes can be viewed as doing "double duty" as domain model validation activities. There is, in fact, a complementary development-validation relationship among all these activities, implying significant interplay and iteration among them in practice.

The Validate Domain Model IDEF₀ diagram is shown in Figure 44. Validate Domain Model includes the Perform Model Walkthrough, Review Domain Model, Validate Domain Model, and Analyze Results processes.

- **Perform Model Walkthrough** processes perform model validation as the domain model is built through team reviews.
- **Review Domain Model** processes review the domain model. Experts within the domain can often discover omissions or errors in the domain model through the review process.

Figure 44: Validate Domain Model IDEF₀ Diagram

- **Validate Domain Model** processes perform formal validation of the domain model through formal mapping back to descriptive domain models and current systems in the domain. Model validation may also involve pilot efforts to assess the practical implications of the prescriptive models on domain architectures, generators, and components.
- **Analyze Results** processes perform the final analysis of whether the domain model is fit for use as a prescriptive model within the domain.

3.4.2.2 Develop Software Architecture

The goal of Software Architecture Development is to produce a *domain architecture model* that encompasses a set of domain architectures, possibly at varying levels of abstraction, that can be used as implementation frameworks in constructing systems from domain assets. The domain architecture model could be represented in a number of ways. These could range from (a) a single "one size fits all" generic architecture, to (b) an enumeration of a collection of such generic architectures, with accompanying decision criteria, to (c) a formal

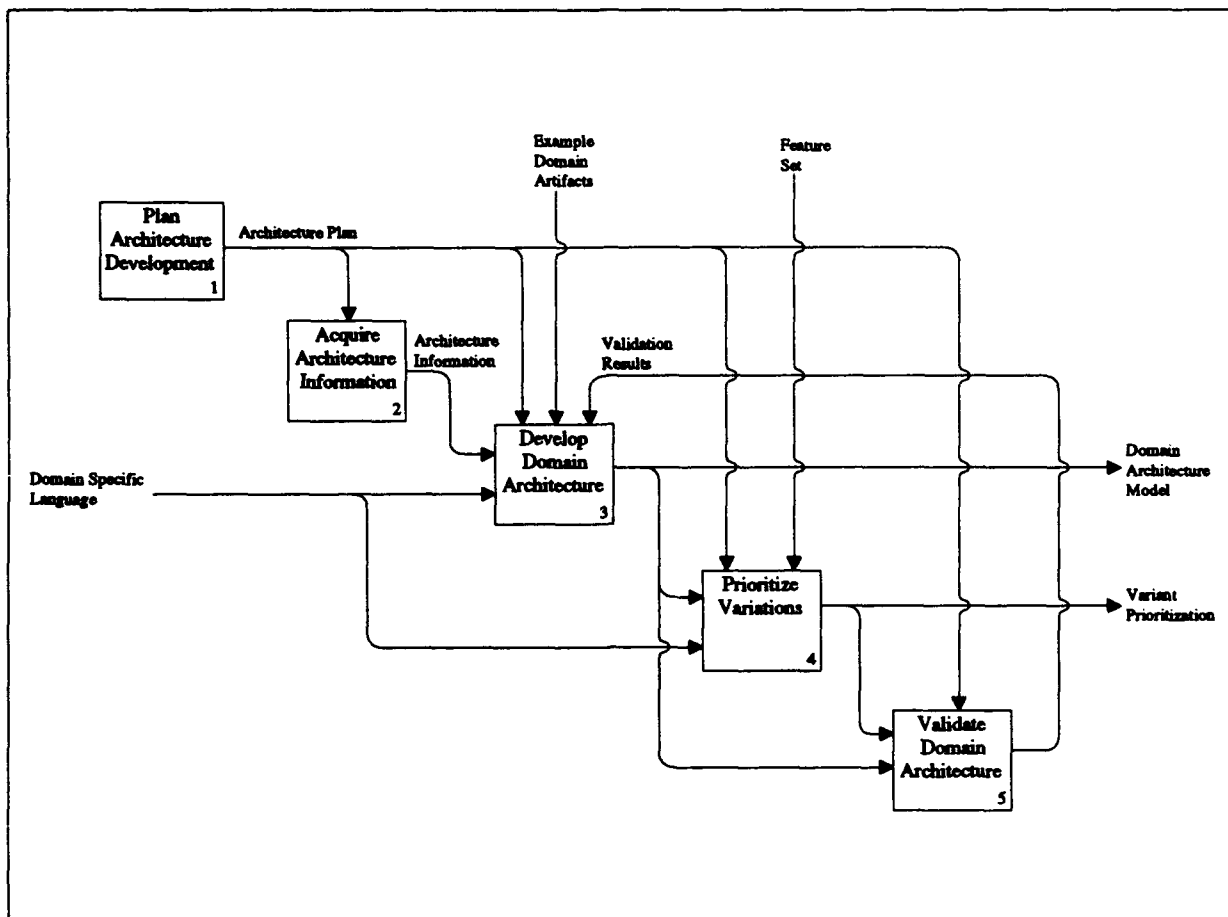


Figure 45: Develop Software Architecture IDEF₀ Diagram

model of the commonality and variability of the potential architectures in the domain that can enable instantiation and tailoring of specific architectures to satisfy specific application constraints.

The Develop Software Architecture IDEF₀ diagram is shown in Figure 45. Develop Software Architecture includes the Plan Architecture Development, Acquire Architecture Information, Develop Domain Architecture, Prioritize Variations, and Validate Domain Architecture processes.

- **Plan Architecture Development** processes determine the objectives and constraints on the architecture development activity. Domain risks are assessed and a risk mitigation plan developed. A validation plan is also developed for the domain architecture.
- **Acquire Architecture Information** processes acquire information needed to develop the architecture model. Information can be acquired through interviews with domain experts, reverse engineering, and design recovery techniques.
- **Develop Domain Architecture** processes develop the domain architecture model.

In general, the model is derived from domain requirements identified in the domain requirements model, and traceability among the models should be established. Ideally, the model should address the full range of variability of architectures in the solution space to maximize the model's applicability, but specific circumstances may significantly restrict the envisioned applicability, and in some cases a single generic architecture may be viewed as appropriate. Domain architecture modeling can often benefit from the development of system prototypes to provide insight into the practicality of certain approaches.

- **Prioritize Variations** processes determine the situations appropriate for the use of particular domain architecture variations. As is generally true of all assets, reuse guidelines should be developed to assist users in instantiating and tailoring domain architectures in specific system contexts.
- **Validate Domain Architecture** processes include performing team walkthroughs and expert reviews of the architecture, as well as assessments in the context of other asset creation activities (see the discussion of the Validate Domain Model processes above).

3.4.2.3 Develop Application Generators

The goal of the Develop Application Generators processes is to produce tools that accept specifications of needed application functionality (i.e., requirements) in terms that are native to the domain and generate application software (sub)systems that provide the desired functionality.

The generated (sub)systems should be consistent with a system architecture derived from the domain architecture model, and the Develop Application Generators processes should thus target development efforts towards some particular set of architectural contexts defined in the domain architecture model. These kinds of development decisions should be captured in a *domain implementation model* that describes how software components and application generators map to the domain architecture and domain requirements models. The ROSE PM does not define a specific activity to develop a domain implementation model; it is currently considered to be implicit in the Map Application Generators to Architecture and Map Components to Architecture processes described below. However, future versions of the ROSE model will likely make this activity more explicit, and organizations that apply the ROSE model should consider defining such an activity explicitly.

The Develop Application Generators IDEF₀ diagram is shown in Figure 46. Develop Application Generators includes the Plan Application Generator Development, Develop Application Generator Test Plan, Develop Generators, Map Application Generators to Architecture, and Validate Application Generators processes.

- **Plan Application Generator Development** processes establish the requirements,

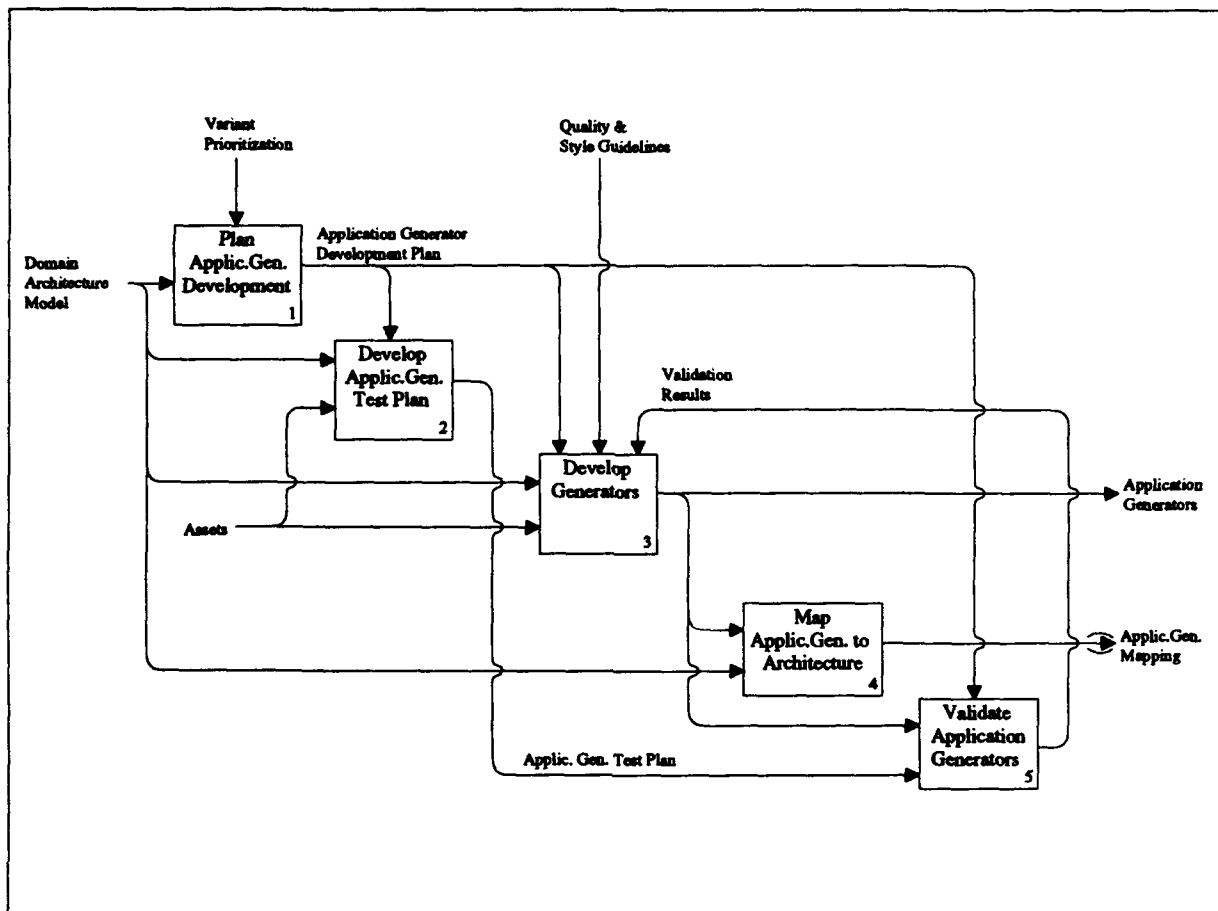
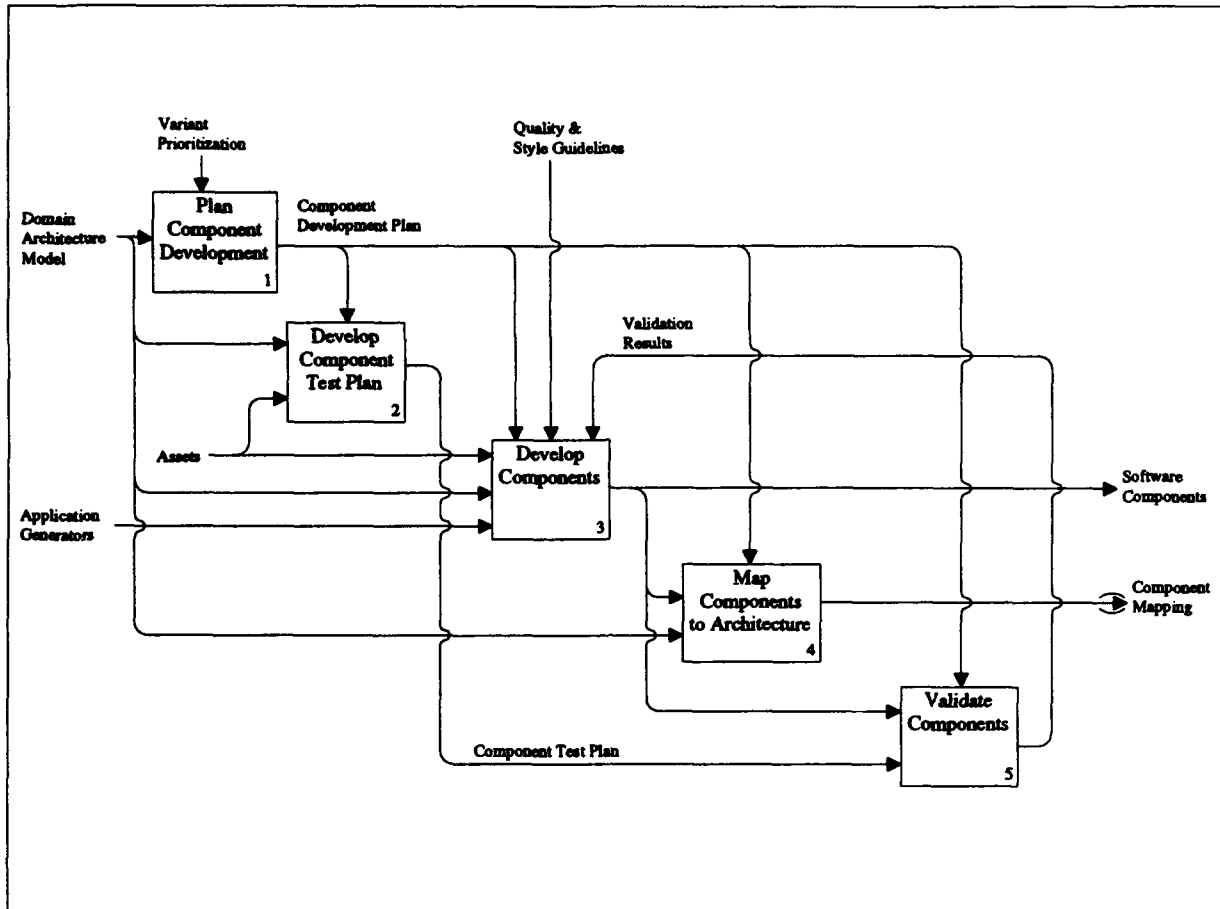


Figure 46: Develop Application Generators IDEF₀ Diagram

context, and decision criteria for application generator development. A risk mitigation plan for the application generators and generator development guidelines are also created.

- **Develop Application Generator Test Plan** processes develop test plans to validate application generators against their requirements.
- **Develop Generators** processes develop and evolve application generators within the domain. Generator usage principles are also developed to guide users on appropriate uses of the generators. Reuse guidelines are developed.
- **Map Application Generators to Architecture** processes map the relationship between the application generators and the domain architecture model.
- **Validate Application Generators** processes perform generator walkthroughs, conduct expert reviews, and test the application generators. In addition, validation may be done in the context of other asset creation activities (see the discussion of the Validate Domain Model processes above for further information).

Figure 47: Develop Software Components IDEF₀ Diagram

3.4.2.4 Develop Software Components

The goal of Develop Software Component processes is to develop a set of reusable software components that implements a specific set of architectural elements, as defined in the domain architecture model. This is generally done by developing the components from scratch or through “reuse-based” reengineering of legacy components, as discussed in the CFRP Definition document.

The Develop Software Components IDEF₀ diagram is shown in Figure 47. Develop Software Components includes the Plan Component Development, Develop Component Test Plan, Develop Components, Map Components to Architecture, and Validate Components processes.

- **Plan Component Development** processes establish the requirements, context, and decision criteria for component development. A risk mitigation plan for the software components and component development guidelines are also created.
- **Develop Component Test Plan** processes develop test plans to validate software

components against their requirements.

- **Develop Components** processes develop and evolve components with the appropriate set of capabilities. Component usage principles are also developed to guide users on appropriate uses of the components. Reuse guidelines are developed.
- **Map Components to Architecture** processes map the relationship between the application generators and domain architecture model.
- **Validate Components** processes perform software component walkthroughs, conduct expert reviews of the components, and test the components. In addition, validation may be done in the context of other asset creation activities (see the discussion of the Validate Domain Model processes above for further information).

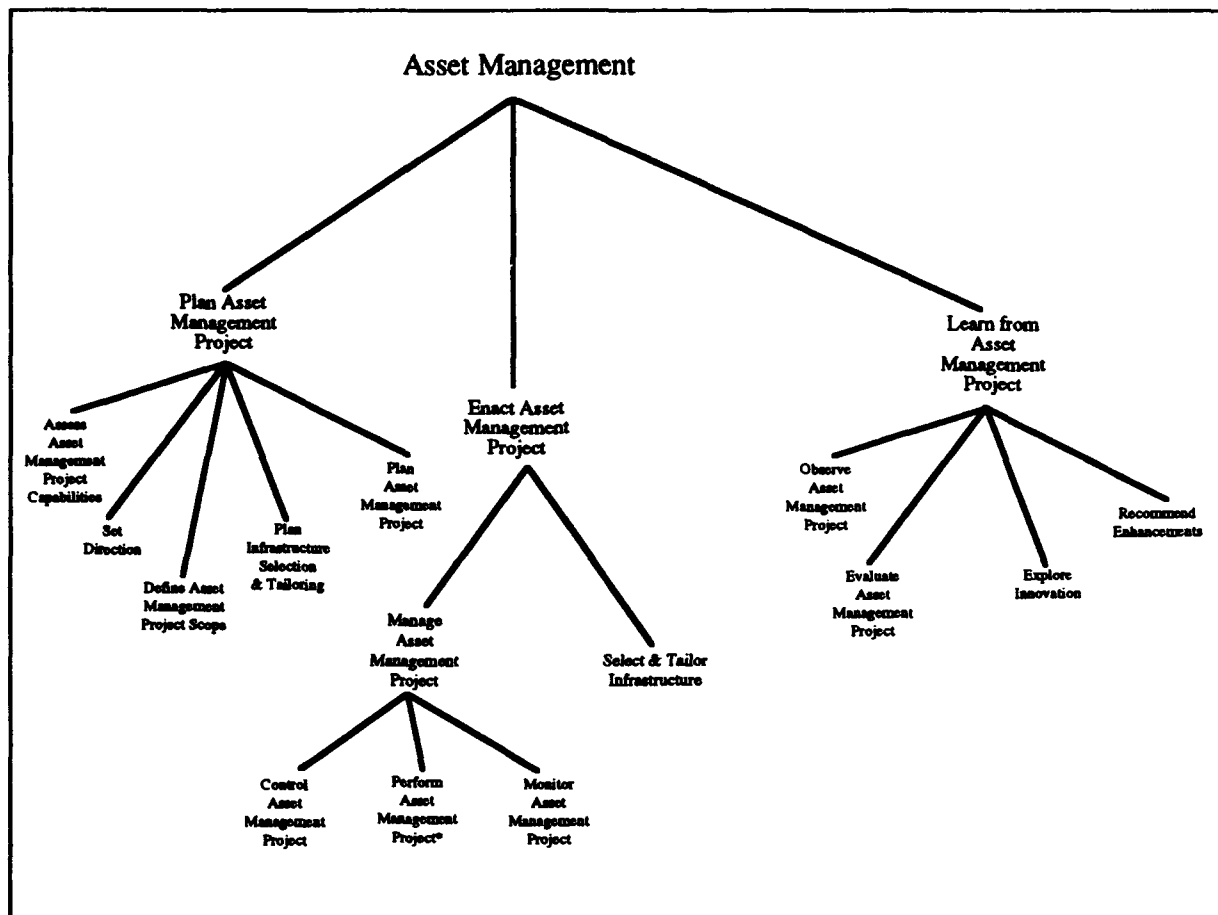


Figure 48: Asset Management Process Abstraction Hierarchy

3.5 Asset Management

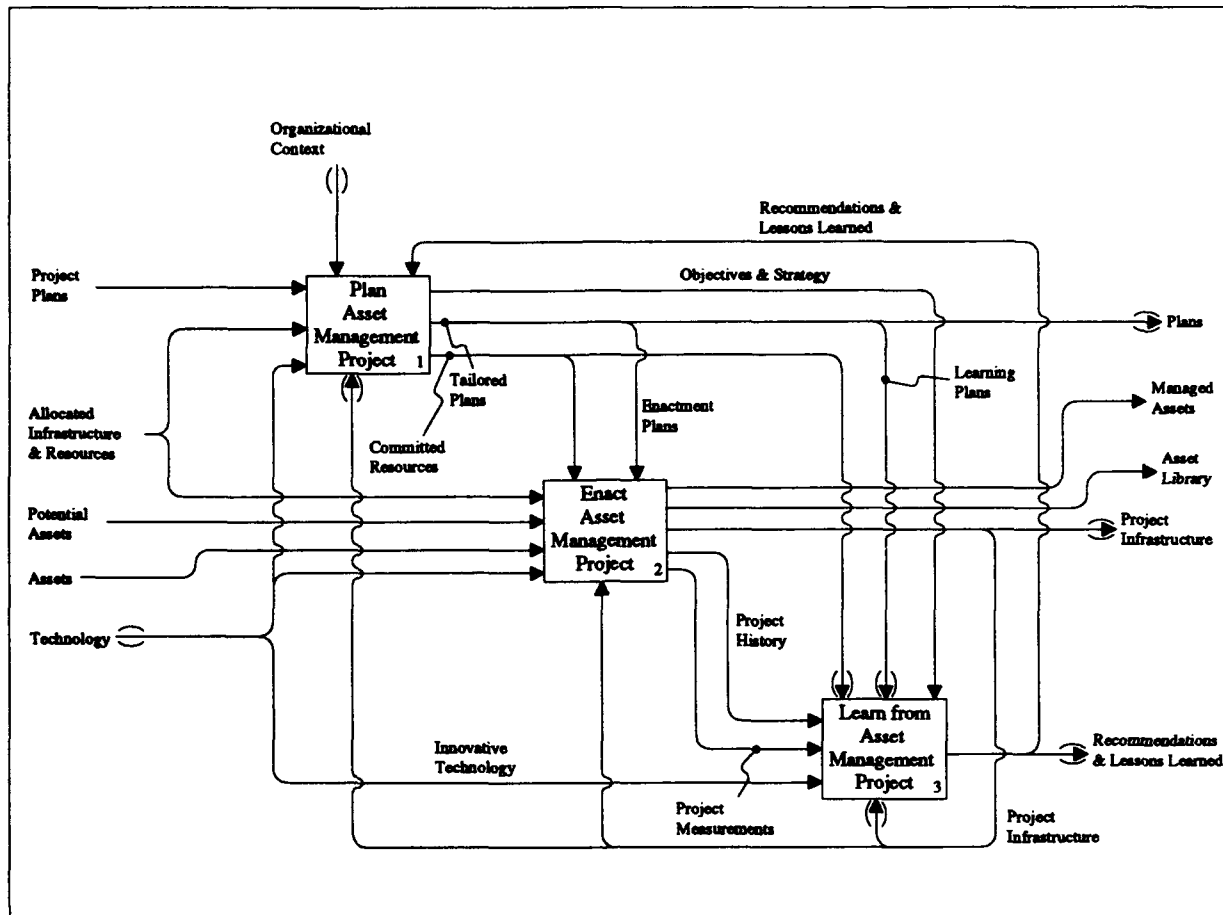
The goal of the ROSE Asset Management project submodel is to acquire, evaluate, and organize assets produced by Domain Engineering processes, and make those assets available as a managed collection that serves as a mediation or brokering mechanism between asset creators and asset utilizers. The process abstraction hierarchy diagram for Asset Management is shown in Figure 48.

Although the term “library” is used throughout this section (and, in fact, throughout the document) to refer to entities that house managed asset collections, this term is not meant (except where noted otherwise) to connote any particular technological approach. A library need not even be automated to effectively manage a collection of assets and serve a useful mediator role between Domain Engineering and Application Engineering processes.

Figure 49 shows a table of the activities within Asset Management. The activities are broken down into Plan, Enact, and Learn process families. Project Performance is part of Enact, so the Project Performance process categories are embedded in the Enact process family in the table. The Asset Management Project Performance process categories are Develop Library

Category	Plan	Enact	Learn
Plan	Review Learning Insights Assess Asset Management Project Capabilities Set Direction Objectives & Strategy	Define Asset Mgmt. Project Scope Plan Infrastructure Selection & Tailoring Plan Process Selection & Tailoring Plan Library Policies & Standards Plan Training	Plan Asset Mgmt. Project Identify Constraints Identify Risks
Enact	Manage Asset Mgmt. Project Control Asset Mgmt. Project Perform Asset Mgmt. Project Monitor Asset Mgmt. Project	Select & Tailor Infrastructure	
1 Develop Library Data Modeling	Determine Data Modeling Approach	Develop Library Taxonomy Select Library Mechanisms Establish Supplemental Tool Reqmts. for Storing & Utilizing Assets Establish Supplemental Tool Capabilities Implement Library Data Model Implementation Navigation & Access Implementation Develop Forms & Metrics Library Metrics Development User Feedback Forms Asset Submittal Forms	Assess Model against Objectives Assess usage against needs Explore alternative data modeling approaches Recommend Changes
2 Operate Library	Plan Library Operations Library Management Policy Constraints Library Operation Goals Reward System Sales Media Coordination with Asset Creators	Perform Library Support Control Library Access Perform Library Configuration Mgmt Perform Asset Interchange Demonstrate Assets	Review Library Operations Status Meetings Quality Management & Improvement
3 Collect Library Metrics	Select Library Metrics Set	Apply Metrics to Library	Analyze Library Measurements
4 Support Assets	Acquire Assets Develop Acquisition Plan Accept Assets Determine Acceptance Criteria Policy Constraints Legal Constraints Domain-Specific Constraints Collect Asset Metrics Select Metrics Set Certify Assets Determine Certification Criteria	Acquire Assets Locate Asset Sources Select Asset Retrieve Assets Accept Assets Determine Adherence to Constraints Policy Constraints Legal Constraints Domain-Specific Constraints Determine Asset Suitability Catalog Assets Classify Assets Describe Assets Install Assets Collect Asset Metrics Apply Metrics to Assets Certify Assets Apply Criteria to Assets	 Accept Assets Perform RCCB Review Catalog Asset Review Cataloged Asset Collect Asset Metrics Analyze Asset Measurements Update Asset Descriptions Certify Assets Analyze Certification Results Update Asset Descriptions
Learn	Observe Asset Mgmt Project	Evaluate Asset Mgmt Project	Explore Innovation Recommend Enhancements

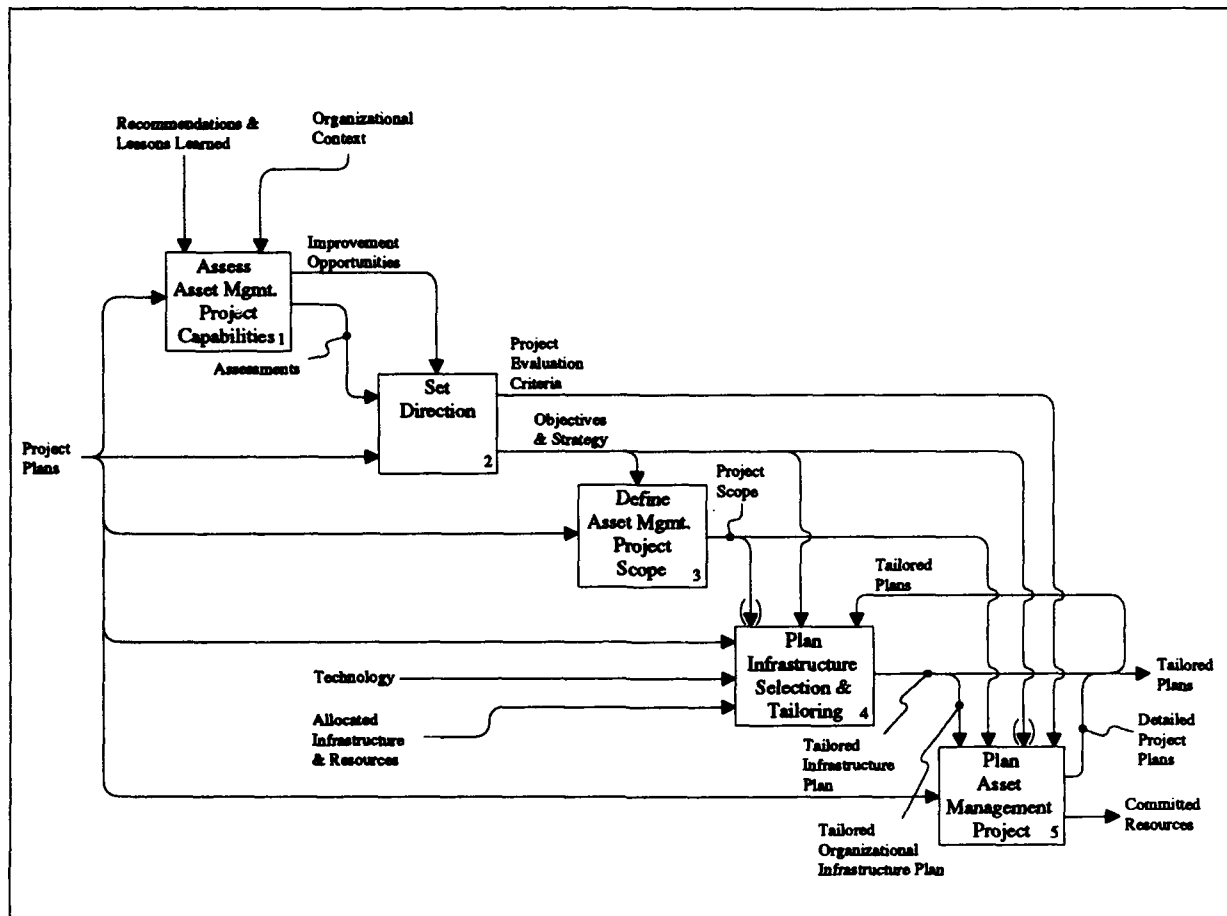
Figure 49: Asset Management Table

Figure 50: Asset Management IDEF₀ Diagram

Data Model, Operate Library, Collect Library Metrics, and Support Assets.

Another view of the Asset Management activities, showing the flow of information among them, can be seen in the IDEF₀ diagrams. Figure 50 shows the top-level IDEF₀ Diagram for Asset Management. This diagram shows that the **Asset Management Process** is broken hierarchically into the **Plan Asset Management Project**, **Enact Asset Management Project**, and **Learn from Asset Management Project** Processes.

Asset Management inputs include **Assets** from Domain Engineering projects that are ready to be evaluated and cataloged; **Potential Assets** developed by other organizations that will be evaluated for suitability for inclusion in the asset library; **Project Plans** inherited from organizational program planning; **Allocated Infrastructure and Resources** that can be tailored to the project; and **Technology** that can contribute to the project's infrastructure and can be applied to automate processes. Controls on Asset Management include **Organizational Context** that guides and constrains various aspects of the project to ensure that they are consistent with overall organization strategies, policies, etc. Outputs from Asset Management include project **Plans**, the tailored **Project Infrastructure**, **Recommendations and Lessons Learned** from project management; **Managed Assets** in the library

Figure 51: Plan Asset Management Project IDEF₀ Diagram

that will be used to construct solution systems in the domain; and an **Asset Library** that provides an organizing scheme for a set of assets and capabilities for accessing and processing the assets.

The following subsections describe the activities carried out in an Asset Management Project. These are divided into the Plan-Enact-Learn project management activities described briefly in Section 3.5.1, and the Asset Management engineering activities (performed within the Enact family) described in Section 3.5.2.

3.5.1 Manage Asset Management

This section describes the processes carried out in managing an Asset Management project. The processes are described briefly since they are generally the same set of processes described in Organization Management, except with a project focus. The differences between Asset Management project management and Organization Management are stressed in this section.

3.5.1.1 Plan Asset Management Project

Figure 51 contains an IDEF₀ Diagram for the Plan Asset Management Project process family. As shown in the figure, the Plan process consists of the following five process categories:

- **Assess Asset Management Project Capabilities** processes characterize the current state of Asset Management practice and capabilities within the organization. The assessment considers a variety of factors, including asset management experience, domain expertise, available support technology, and availability of assets. Included in this process is a review of recommendations and lessons learned from previous cycles to identify specific improvements that can be made to the organization's Asset Management capabilities.
- **Set Direction** processes determine a strategy and objectives for the Asset Management project. A high-level strategy may be defined in the Plan Projects process category within Organization Management. In Set Direction, a more detailed strategy and objectives are defined, based on the high-level strategy. Once a strategy and objectives have been developed, success criteria are identified to determine whether the objectives are met.
- **Define Asset Management Project Scope** processes bound the scope of key aspects of the project. This may include defining the boundaries of the domain(s) to be managed in more detail than was done in the Scope Line of Business process in Organization Management. It may also involve constraining a variety of other aspects of the project, such as desired levels of asset quality, library accessibility and interoperability, etc.
- **Plan Infrastructure Selection and Tailoring** processes plan the technical, organizational, and educational infrastructure needs of the project. These needs can be satisfied by the infrastructure at the organizational level, tailored from the infrastructure at the organizational level, or, if necessary, created specifically for the project. Project infrastructure planning includes plans for processes, policies, standards, and training.
- The **Plan Asset Management Project** processes perform detailed project planning. Detailed project planning includes the development of a schedule, budget, and resource needs based on the high-level schedule and budget developed in the Plan Projects process category in Organization Management. Political, technical, schedule, and cost constraints on the project are identified and control mechanisms are planned. Project risks are identified and evaluated as "high", "medium", or "low". Risk mitigation plans and mechanisms are developed for handling high risks.

3.5.1.2 Enact Asset Management Project

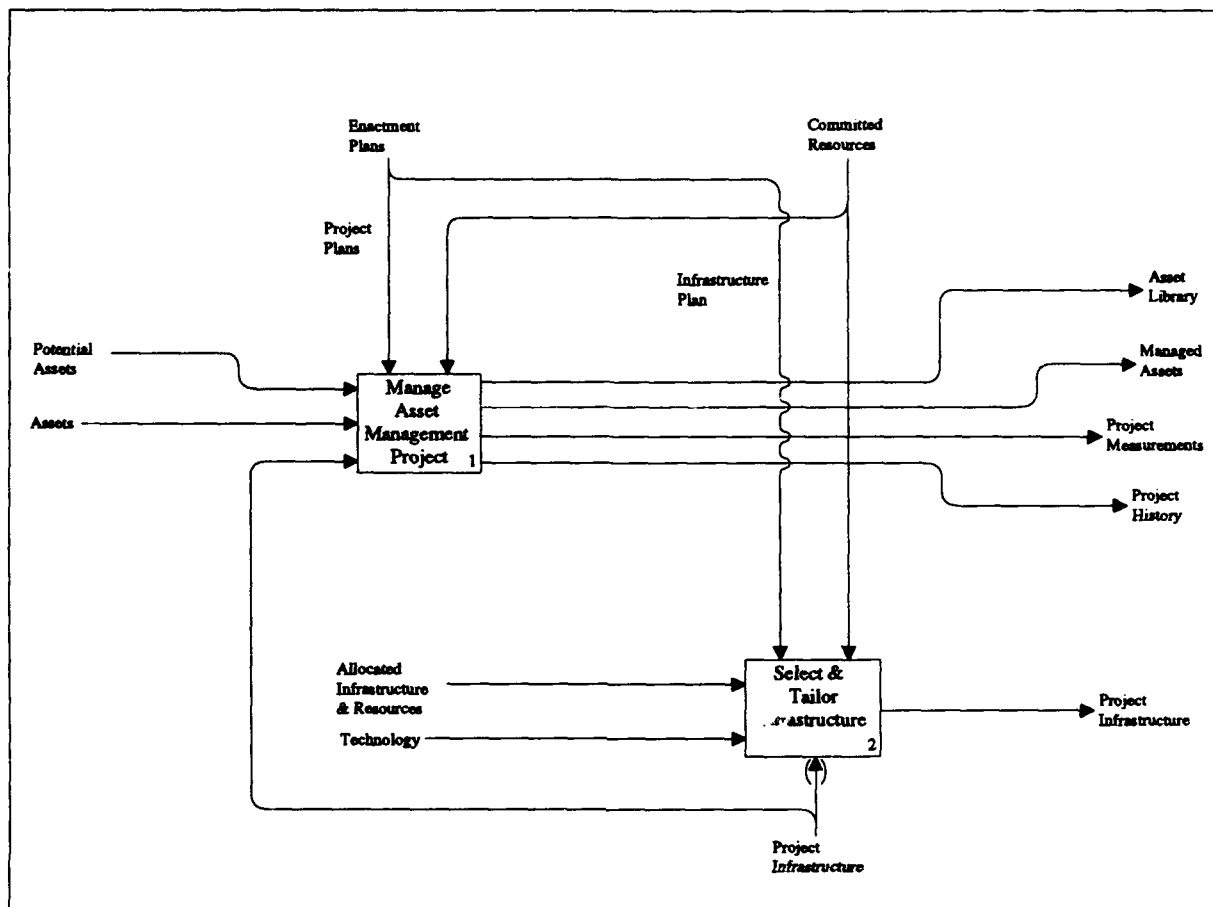


Figure 52: Enact Asset Management Project IDEF₀ Diagram

The Enact Asset Management Project process family manages the day-to-day activity of the project and ensures that an infrastructure sufficient to meet the needs of the project is established and maintained.

Figure 52 contains an IDEF₀ diagram for the Enact Asset Management Project process family. As shown in the figure, the Enact process family consists of the **Manage Asset Management Project** and **Select and Tailor Infrastructure** process categories.

Manage Asset Management Project

The Manage Asset Management Project process category establishes a temporal context for the performance of the Asset Management project and performs detailed project supervision. The Manage Asset Management Project IDEF₀ diagram is shown in Figure 53. The Manage Asset Management Project process includes Control Asset Management Project, Perform Asset Management Project, and Monitor Asset Management Project.

The Control Asset Management Project process activities intervene with project per-

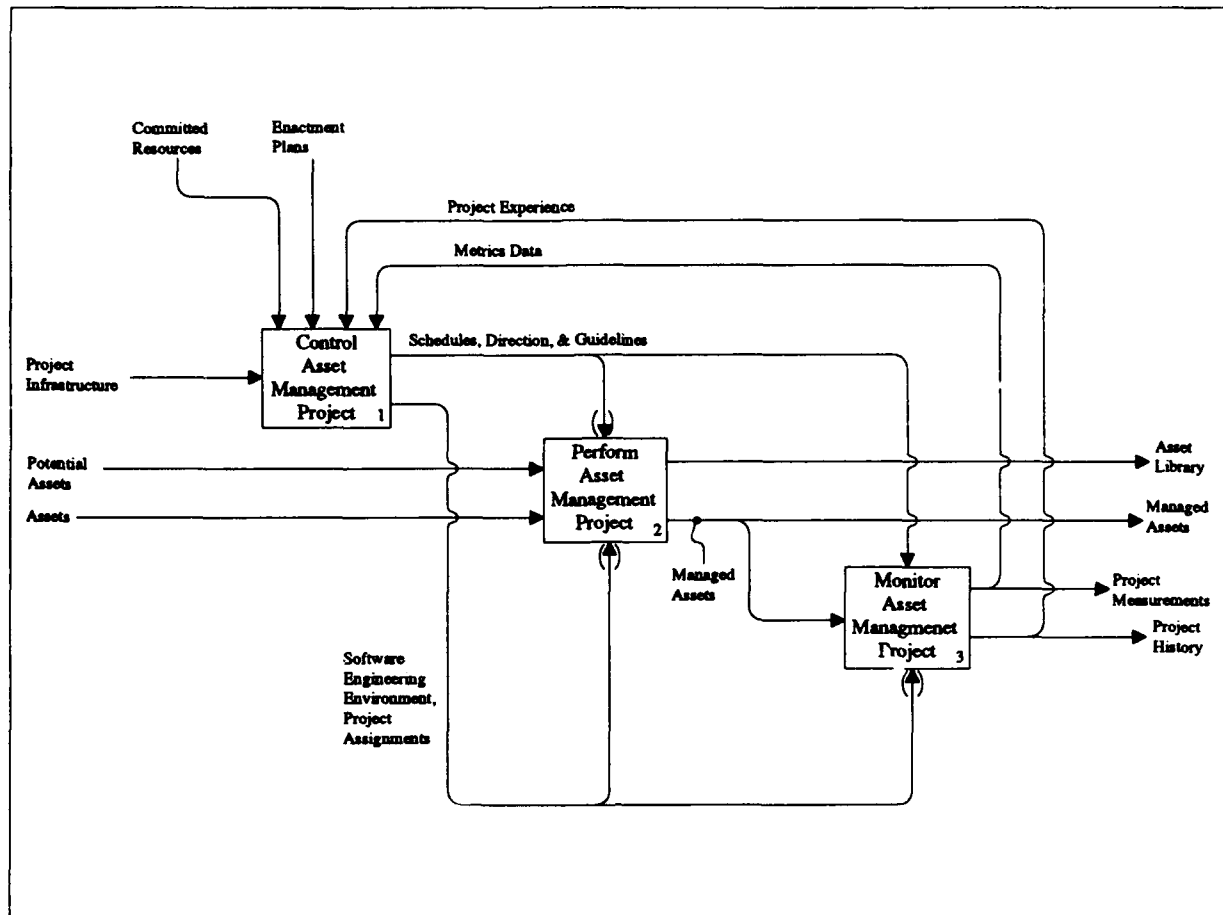


Figure 53: Manage Asset Management Project IDEF₀ Diagram

formance in order to keep the project on track relative to objectives set during Organizational Planning. Project control is the “management” function (in the most conventional sense) for the Asset Management project. The goal of project control activities is to optimize overall project performance, as measured by factors such as the rate of growth of the managed asset population, the degree to which the assets are utilized, the overall quality of the assets and resulting systems, the effectiveness of the infrastructure, etc.

The **Perform Asset Management Project** activities at the core of Enactment are where the engineering processes being enacted are “hooked in” to Asset Management project management and actually performed by individual staff members. These engineering processes are described in Section 3.5.2. From a management perspective, project performance activities also involve tactical decisions about *how* the work will be performed on a detailed, day-to-day basis. At one level, project performance activities can be viewed as individualized techniques for filling in the minute implementation details that are generally missing from project processes established in Project Planning.

The **Monitor Asset Management Project** activities capture “learning-oriented” information from the Asset Management project as it is performed. Some of this information

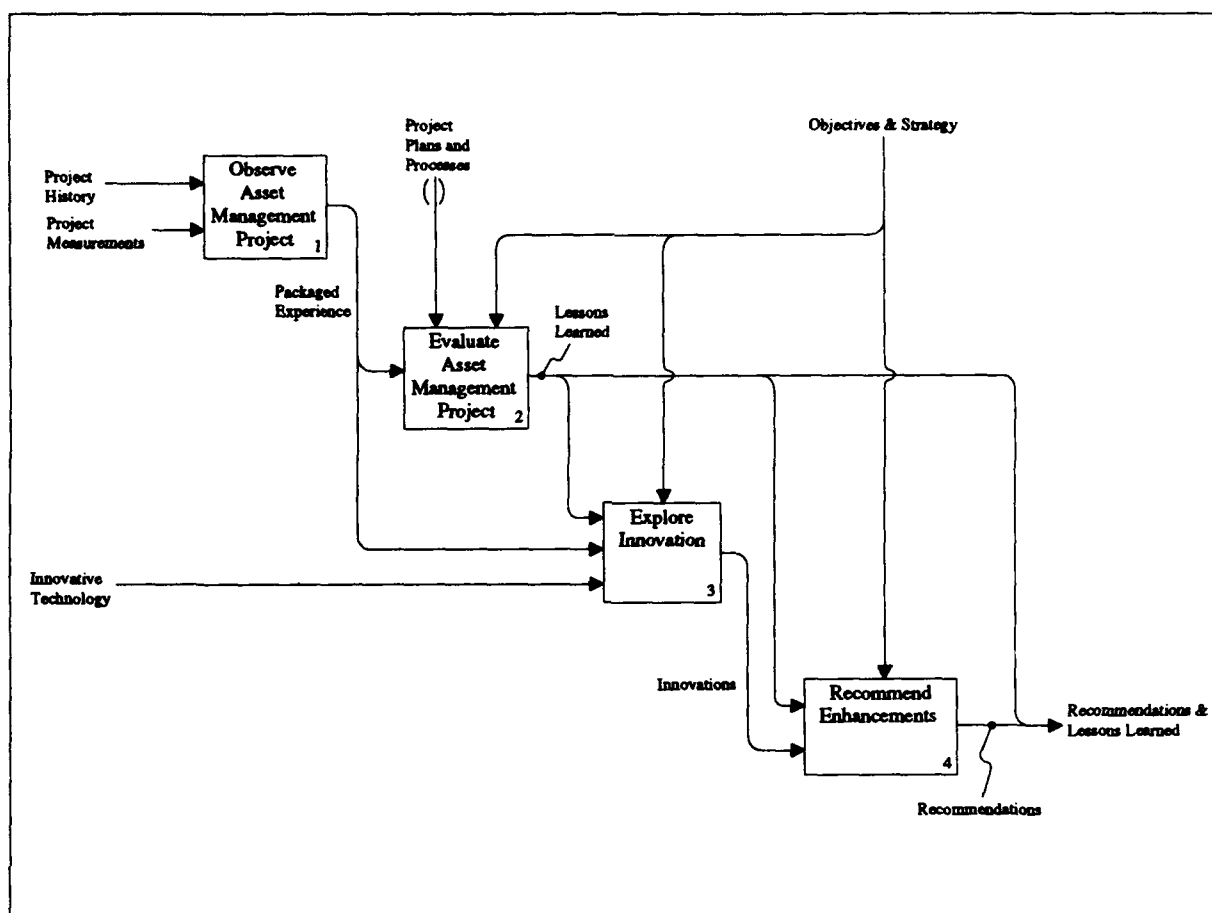


Figure 54: Learn from Asset Management Project IDEF₀ Diagram

provides feedback to project control activities in order to adjust schedules, budgets, milestones, incentives, guidelines, or task assignments. Some of the same information may serve as input to the Learn from Asset Management Project processes, along with data collected in accordance with process and product metrics identified in Plan Asset Management Project, and project history data such as rationale for decisions made and interim work products created.

Select and Tailor Infrastructure

Select and Tailor Infrastructure processes select the infrastructure to meet the project's needs from the organization's infrastructure. This infrastructure may be tailored to the project and missing infrastructure capabilities to satisfy new project-specific needs may be developed.

3.5.1.3 Learn from Asset Management Project

Processes in the Learn from Asset Management Project process family assess the overall

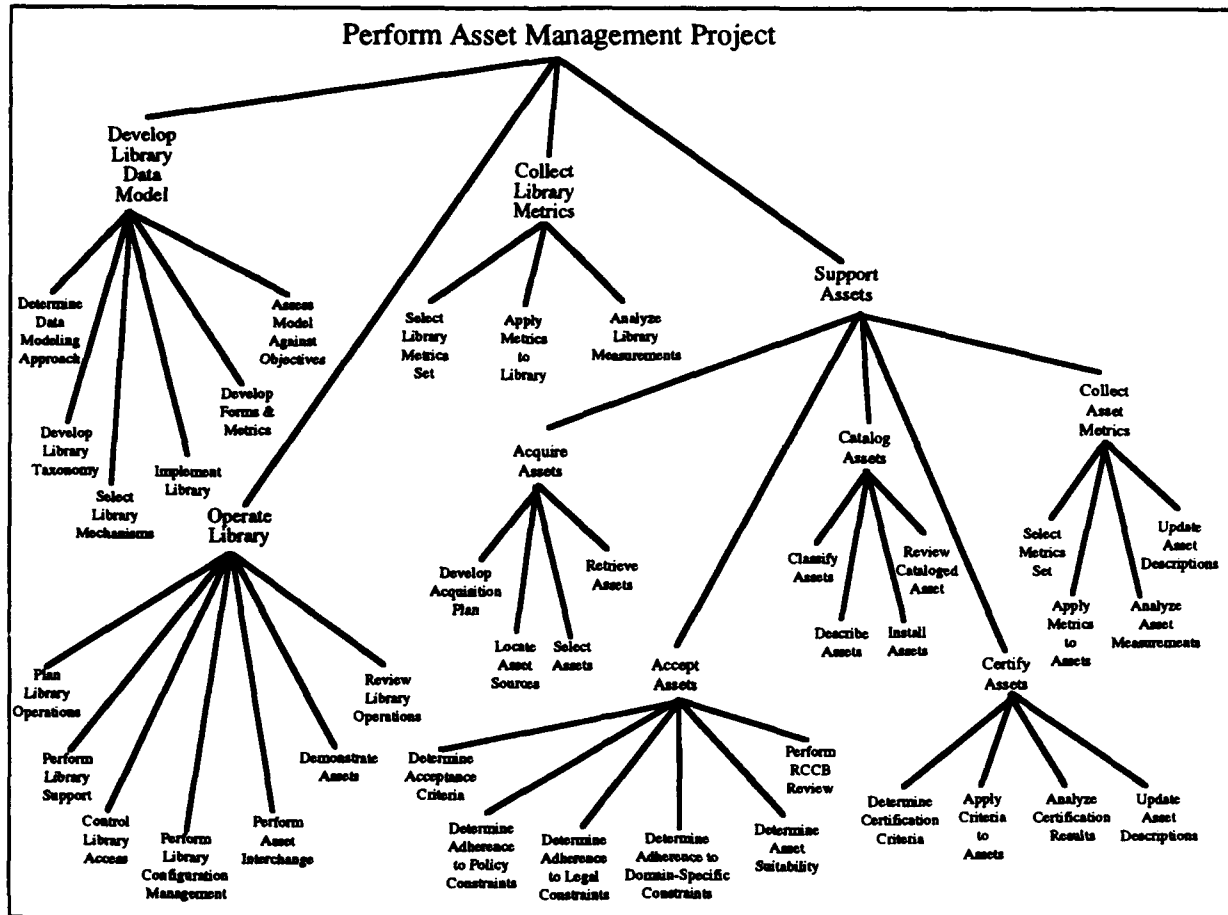


Figure 55: Perform Asset Management Project Process Abstraction Hierarchy

success of the strategies and plans put into place by the Plan Asset Management Project process family, comparing project results and experience against project objectives, to improve the quality of the plans and infrastructure. Figure 54 contains an IDEF₀ diagram for the Learn from Asset Management Project process family. As shown in the figure, the Learn process family consists of **Observe Asset Management Project**, **Evaluate Asset Management Project**, **Explore Innovation**, and **Recommend Enhancements**. These process categories were discussed in section 3.3.3, on Organization Management Learning. Project level learning is similar to organizational learning, except that the focus is on the individual Asset Management project.

3.5.2 Perform Asset Management

The **Perform Asset Management Project** process included among the Enact Asset Management Project processes described above is where the engineering activities enacted in the project are “hooked in” to project management activities. This section describes these engineering activities, divided into process categories that include their own localized

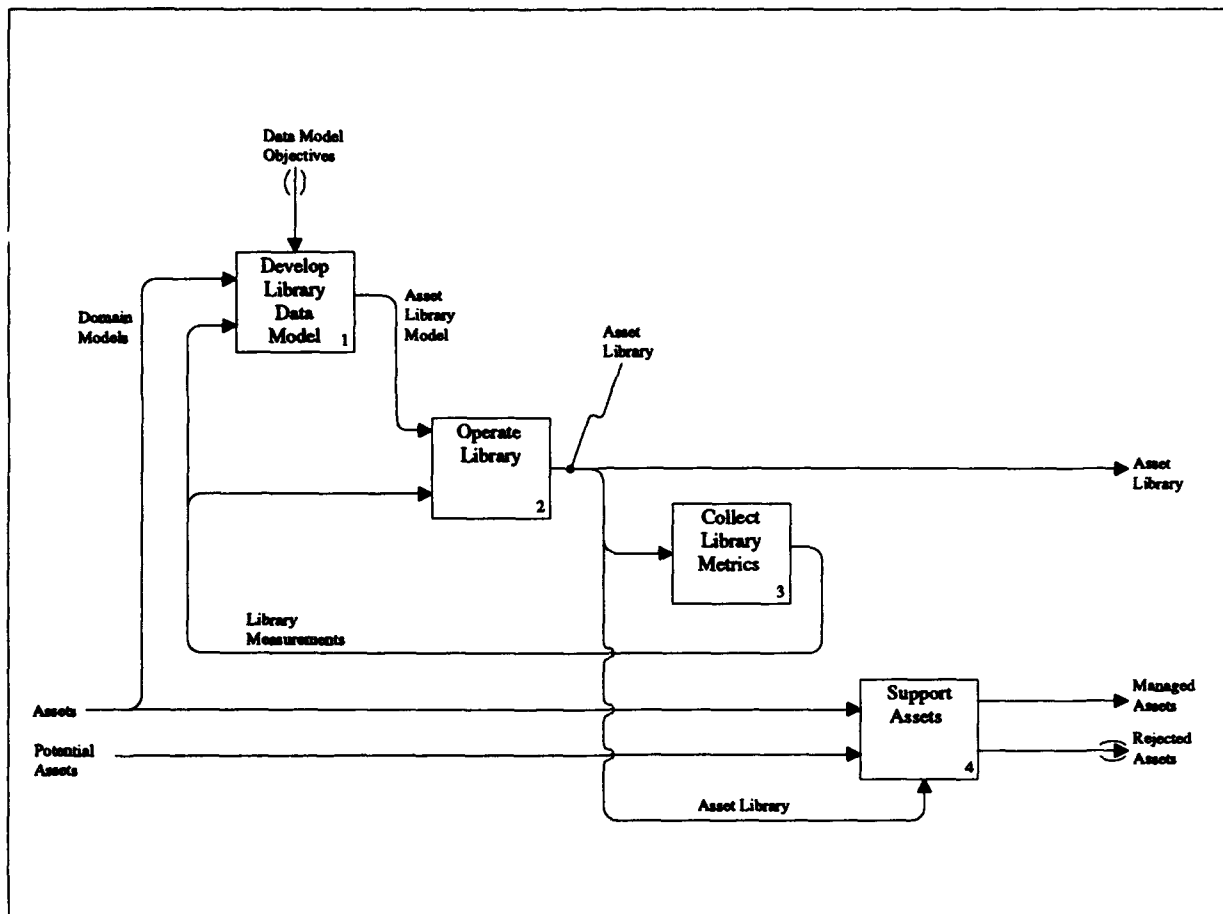


Figure 56: Perform Asset Management Project IDEF₀ Diagram

Plan-Enact-Learn activities. The process abstraction hierarchy diagram for Perform Asset Management Project is shown in Figure 55.

The Perform Asset Management Project IDEF₀ diagram is shown in Figure 56. Perform Asset Management Project includes Develop Library Data Model, Operate Library, Collect Library Metrics, and Support Assets. Each of these process categories is described in detail below.

3.5.2.1 Develop Library Data Model

The goal of Develop Library Data Model processes is to develop a data model for describing assets within a library. This *library data model* synthesizes those aspects of the domain requirements, architecture, and implementation models produced by Domain Engineering that are needed to directly support Application Engineering. Although it is referred to here as a single data model, it may consist of multiple distinct models that are logically related or integrated.

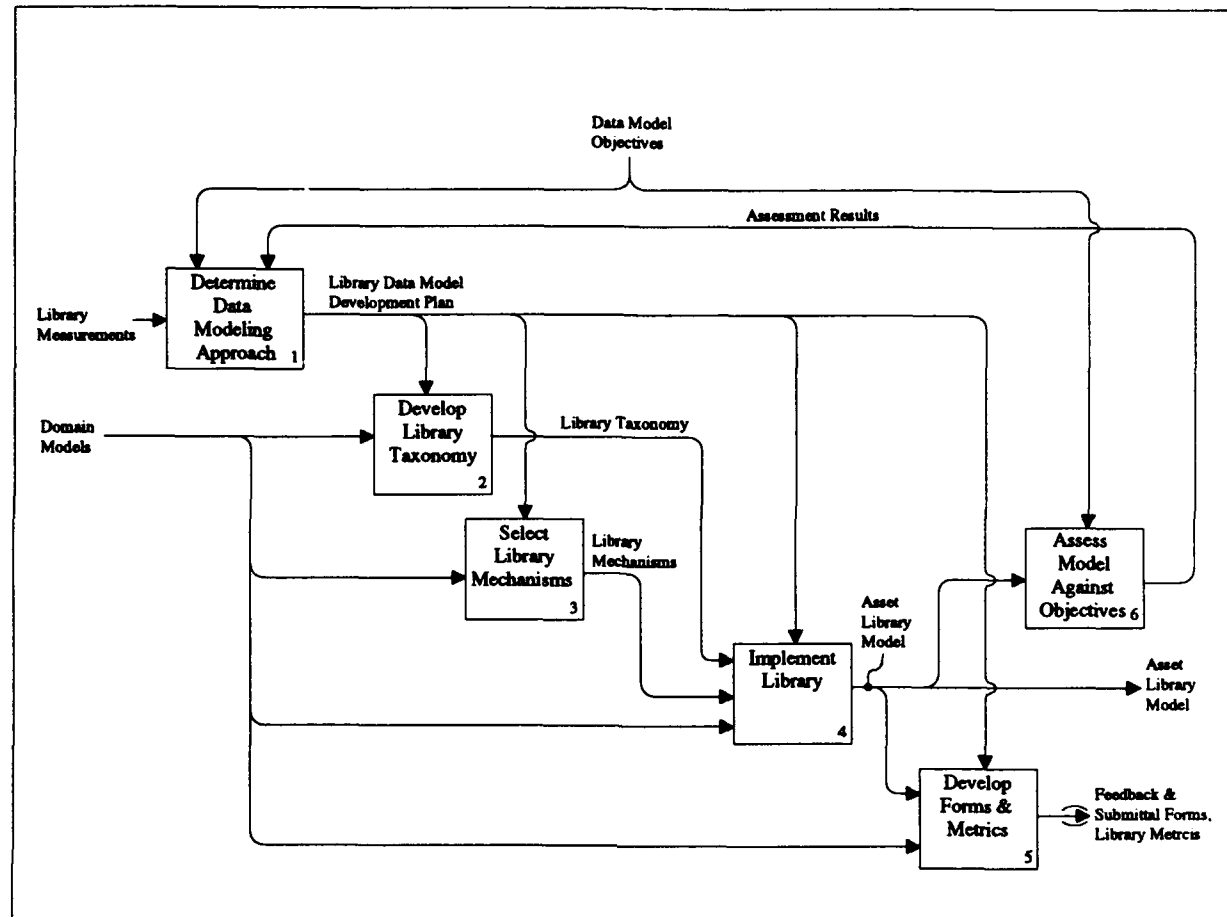


Figure 57: Develop Library Data Model IDEF₀ Diagram

The Develop Library Data Model IDEF₀ diagram is shown in Figure 57. Develop Library Data Model includes the Determine Data Modeling Approach, Develop Library Taxonomy, Select Library Mechanisms, Implement Library, Develop Forms and Metrics, and Assess Model Against Objectives processes.

- **Determine Data Modeling Approach** processes determine the specific approach taken to produce the library data model. This approach is highly dependent on the characteristics of the domain information models produced in Domain Engineering and on the objectives of the library in supporting Application Engineering, as determined by the needs of asset utilizers.
- **Develop Library Taxonomy** processes develop the library data model, possibly consisting of multiple integrated models, in accordance with the selected data modeling approach. The library data models will typically be derived from or strongly related to the domain information models, if they are available, to emphasize aspects of the models that will be most relevant to the asset utilizer. The library data models may also augment the domain information models with information that facilitates management

of the assets or gives the utilizer additional insight into the management or utilization history of the assets.

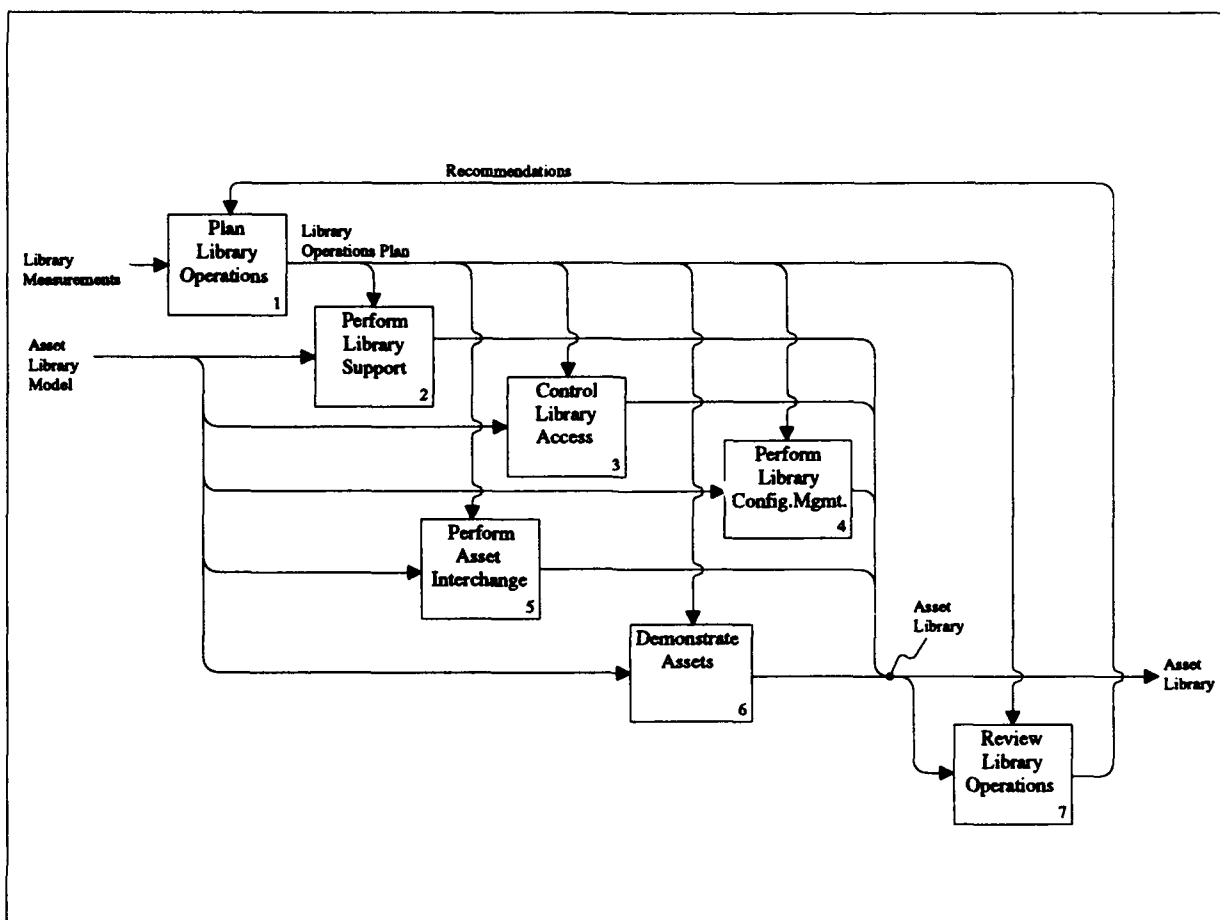
Taxonomic models of one form or another are commonly used to establish a classification scheme that partitions assets into different classes or categories that can be queried or browsed. Other models may be more structural in nature to place assets in their architectural contexts. Library data models can be represented in a variety of ways, including simple hierarchies, faceted schemes, object-oriented class hierarchies, entity-relationship models, and semantic networks. Multiple integrated models provide library users with alternative taxonomic and structural views of the domain and thus present a variety of alternative, yet integrated asset browsing, querying, and retrieval strategies.

- **Select Library Mechanisms** processes select or develop tools that are integrated with the basic library software to provide mechanisms for processing the data model and assets to support Asset Utilization during Application Engineering. These tools provide capabilities for doing things such as: viewing assets, understanding assets, testing assets, executing assets for evaluation, providing views of asset interrelationships and architectural structures, extracting sets of closely related assets, composing assets to build applications, executing application generators to generate portions of an application, etc.
- **Develop Forms and Metrics** processes develop library metrics and user feedback and asset submittal forms.
- **Implement Library** processes finalize the data model and acquire/implement and install the mechanisms to support library usage.
- **Assess Model Against Objectives** processes review the developed library data model to ensure that it meets objectives.

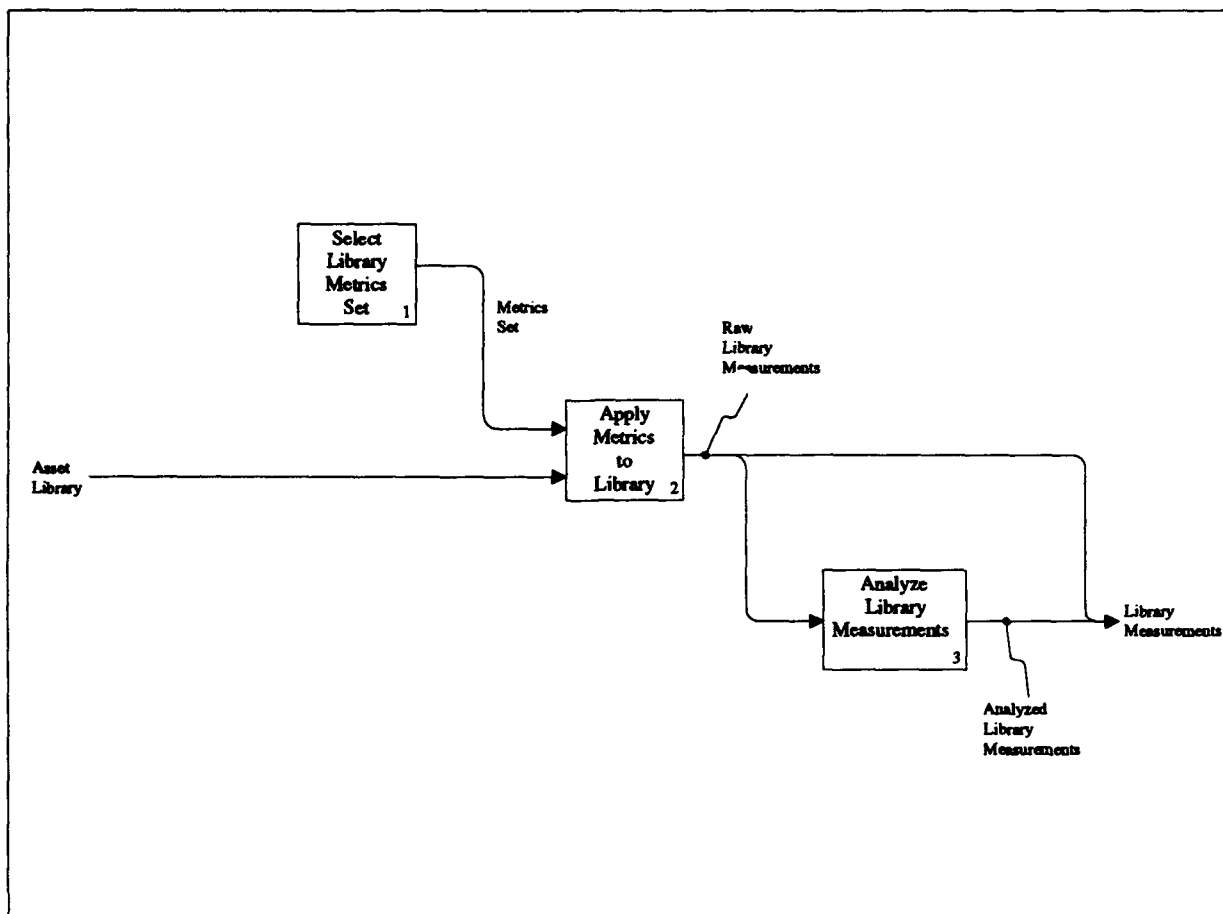
3.5.2.2 Operate Library

Operate Library processes assure the availability and accessibility of the library and its associated assets to library customers. The Operate Library IDEF₀ diagram is shown in Figure 58. Operate Library includes the Plan Library Operations, Perform Library Support, Control Library Access, Perform Library Configuration Management, Perform Asset Interchange, Demonstrate Assets, and Review Library Operations processes.

- **Plan Library Operations** processes provide library planning activities, such as coordination with domain engineers to establish an asset supply and feedback mechanisms; development of Library Operation Goals, Library Operation Policies and Procedures; and development of Incentive Systems and Promotional Material.

Figure 58: Operate Library IDEF₀ Diagram

- **Perform Library Support** processes support the basic operational needs of the asset library including: catalog generation, asset storage and distribution, hot line operation, bulletin board operation, mail system operation, tool acquisition and integration, system installation, administration, and operation, backups, and metrics reporting.
- **Control Library Access** processes restrict access to asset libraries, asset library subdomains, or individual assets. Control functions can include new user registration, access restrictions, password security, session auditing, and security incident response.
- **Perform Library Configuration Management** processes perform configuration management functions associated with asset library operation including managing multiple versions of assets, developing change histories, validating guideline adherence, developing quality and other guidelines, responding to question and problems, and archiving assets no longer used. A related library service is asset subscription, which allows users that "subscribe" to a particular asset to be informed of all changes to that asset (and, optionally, to be given updated versions of it) as it evolves.
- **Perform Asset Interchange** processes define procedures used to interoperate with

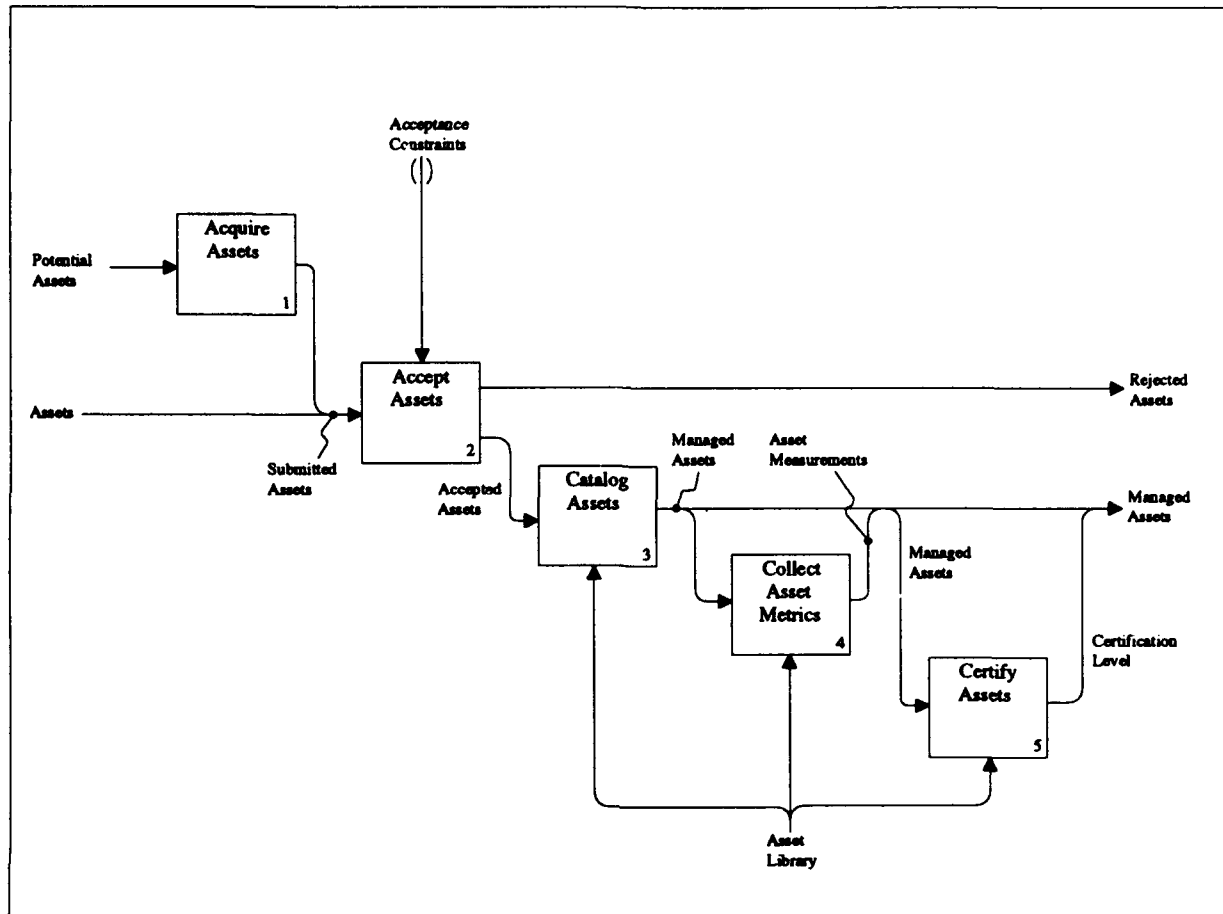
Figure 59: Collect Library Metrics IDEF₀ Diagram

remote asset libraries. Such interoperation may take a variety of forms, including accessing the assets in the remote libraries in a transparent and seamless manner or exporting local assets to those libraries in batch mode.

- **Demonstrate Assets** processes demonstrate the effectiveness of assets to show their benefit to potential asset users.
- **Review Library Operations** processes include holding status meetings and managing and improving the quality of the asset library based on user feedback as well as internal assessment.

3.5.2.3 Collect Library Metrics

The goal of the Collect Library Metrics processes is to collect information that can be used to assess the characteristics and effectiveness of the library overall, rather than of individual assets. Examples of such metrics include number of users, average number of user sessions

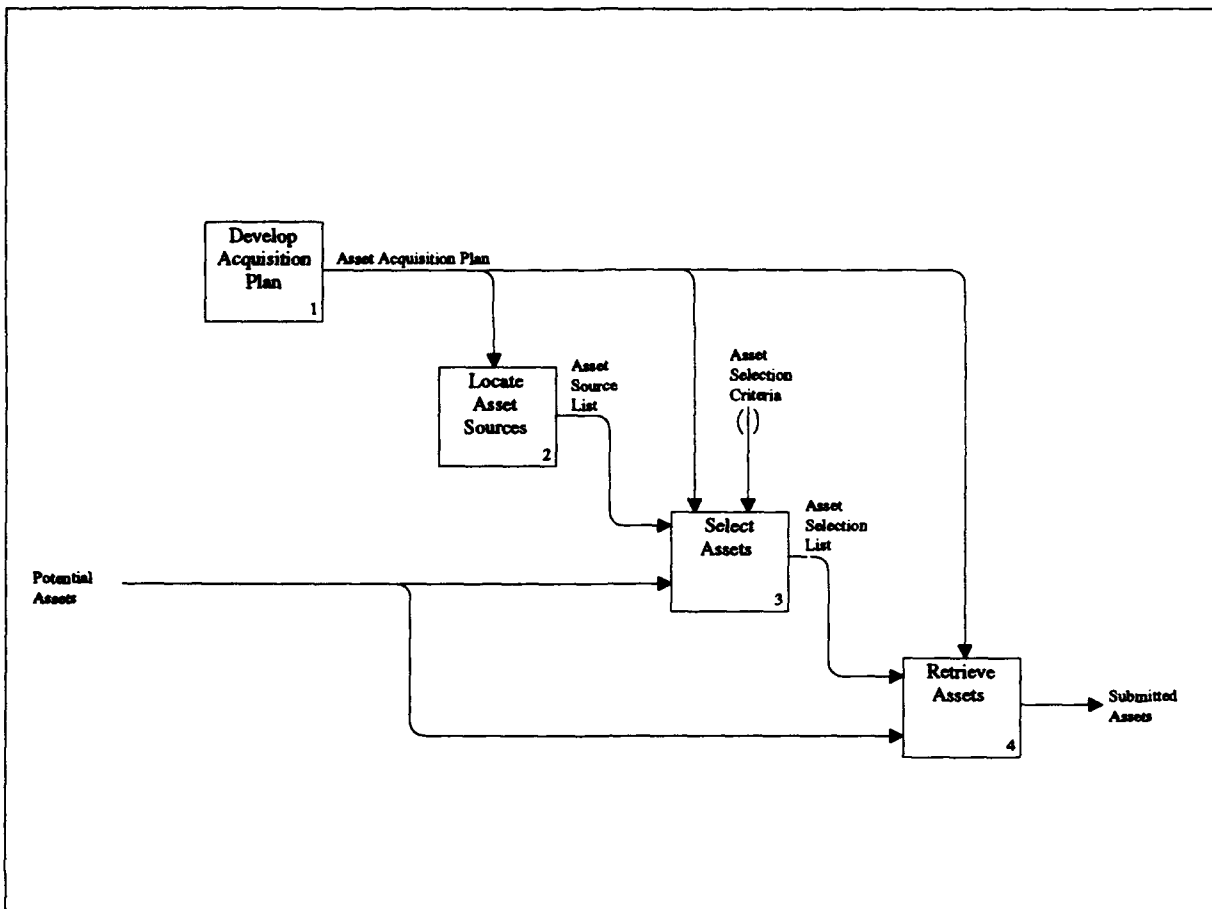
Figure 60: Support Assets IDEF₀ Diagram

per day, number of assets retrieved by users, average certification level of assets, and browsing and query statistics. This metrics information can be used to evolve and improve library management artifacts and library services, or may be provided to Organization Management processes to contribute to overall process measurement and improvement activities within the organization.

The Collect Library Metrics IDEF₀ diagram is shown in Figure 59. Collect Library Metrics includes the Select Library Metrics, Apply Metrics to Library, and Analyze Library Measurements processes.

3.5.2.4 Support Assets

Support Assets processes support the addition of new assets to the asset library, the collection of metrics about library assets, and the certification of library relative to defined criteria. The Support Assets IDEF₀ diagram is shown in Figure 60. Support Assets includes the Acquire Assets, Accept Assets, Catalog Assets, Collect Asset Metrics, and Certify Assets processes.

Figure 61: Acquire Assets IDEF₀ Diagram

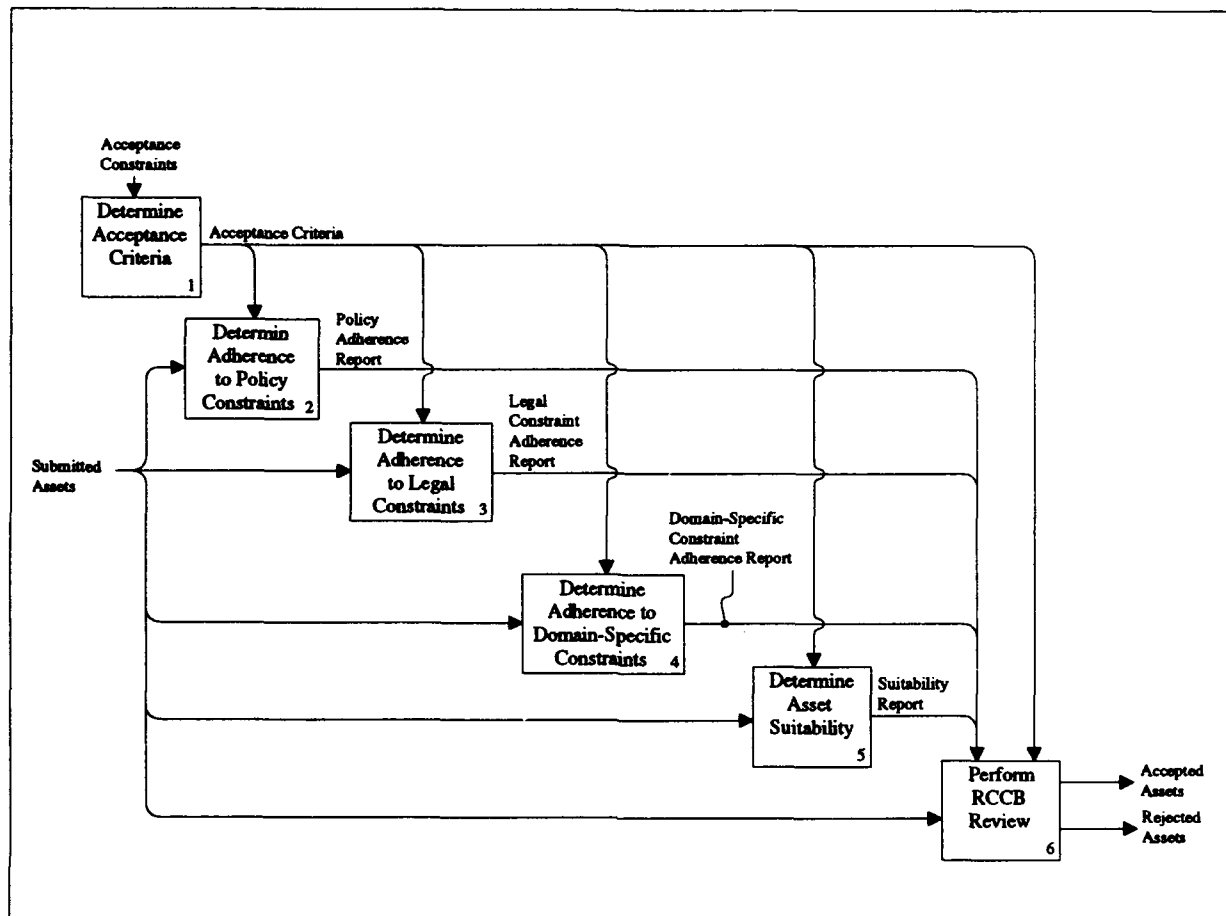
Acquire Assets

The principal goal of the Acquire Assets processes is to obtain assets from internal Domain Engineering projects, external asset libraries, and other sources to support asset utilization. These processes obtain assets that satisfy domain and architectural requirements so that they can be made available to utilizers through a domain-specific library. External asset sources can include external asset libraries, projects developing applications within a relevant domain, commercial or government off-the-shelf products, and external individuals or organizations that voluntarily submit candidate assets.

The Acquire Assets IDEF₀ diagram is shown in Figure 61. Acquire Assets includes the Develop Acquisition Plan, Locate Asset Sources, Select Assets, and Retrieve Assets processes.

Accept Assets

The goal of the Accept Assets processes is to ensure that an asset that is a candidate for inclusion in a library satisfies relevant policy, legal, and domain-specific constraints. The

Figure 62: Accept Assets IDEF₀ Diagram

Accept Assets IDEF₀ diagram is shown in Figure 62. Accept Assets includes the Determine Acceptance Criteria, Determine Adherence to Policy Constraints, Determine Adherence to Legal Constraints, Determine Adherence to Domain-Specific Constraints, Determine Asset Suitability, and Perform RCCB Review processes.

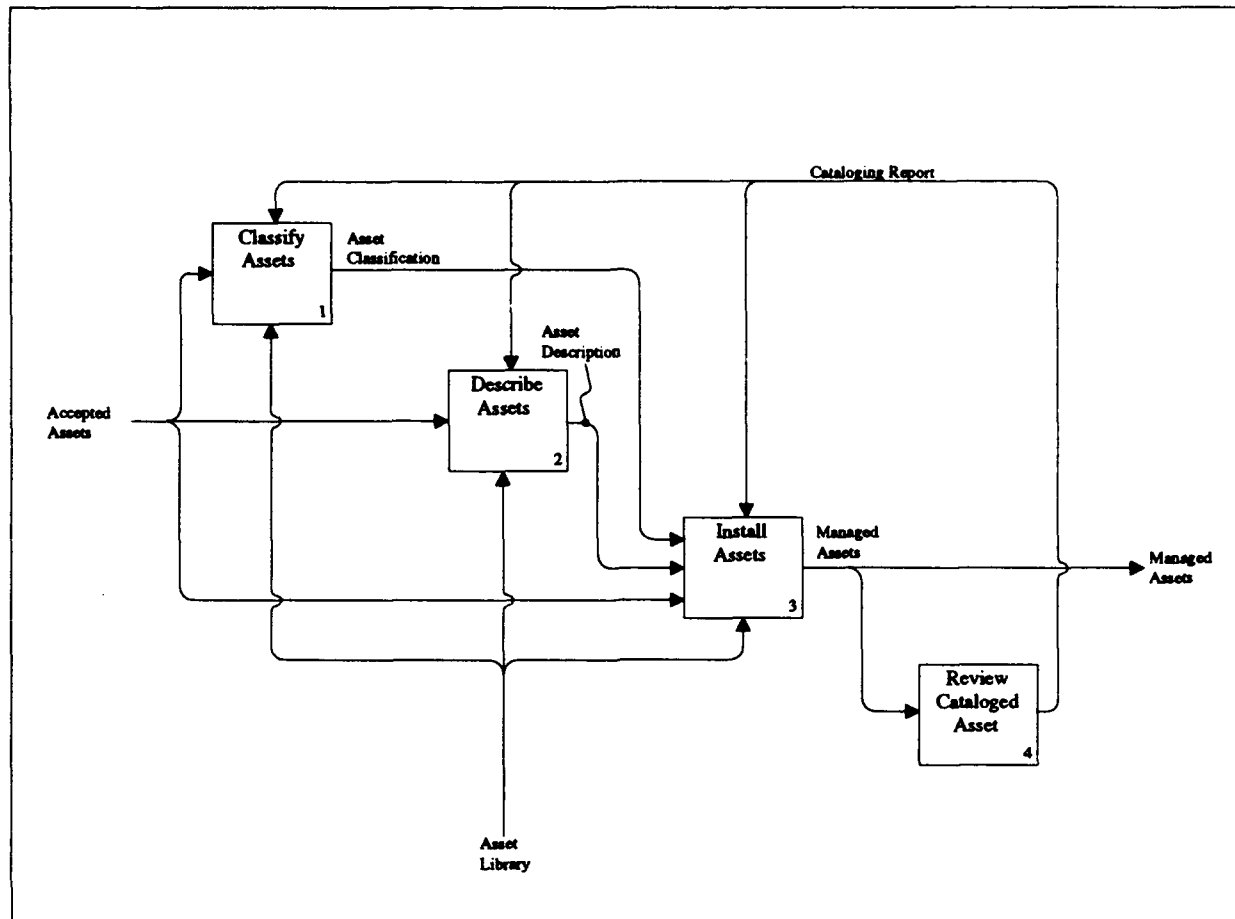
- **Determine Acceptance Criteria** processes determine the specific policy, legal, and domain-specific constraints against which the asset will be checked for adherence.
- **Determine Adherence to Policy Constraints** processes ensure that assets in a library satisfy at least minimal criteria for quality and suitability for use in Asset Utilization activities. Such constraints are generally imposed internally by an organization and often are expressed in terms of requirements on the descriptive information that accompanies a candidate asset, and sometimes in terms of requirements on the asset itself. These constraints may address technical or non-technical issues and may be arbitrarily loose or stringent. Examples of issues that could be addressed include programming language, documentation standards, genealogical information, and information source.

- **Determine Adherence to Legal Constraints** processes focus primarily on legal constraints which restrict the access, distribution, or use of an asset, independent of its perceived technical quality or suitability. Legal constraints are generally imposed by external organizations. Consideration of legal constraints is particularly important for assets acquired from external sources such as public, government-supported, or commercial asset libraries, or sometimes even other projects within the same organization.
- **Determine Adherence to Domain-Specific Constraints** processes assess assets against domain-specific asset acceptance constraints defined by the domain requirements, architecture, and implementation models produced by Domain Engineering processes. These constraints establish qualification criteria with respect to the functions, interfaces, interactions, side effects, performance, etc. (sometimes called the "form, fit, and function") of candidate assets relative to domain needs.
- **Determine Asset Suitability** processes review the results of determining asset adherence to constraints and make specific recommendations about whether an asset should be accepted for inclusion into the library or not.
- **Perform RCCB Review** processes make the final determination about asset inclusion or exclusion from the asset library. The Reuse Configuration Control Board (RCCB) may be composed of a wide variety of people, including library staff members, representatives from Domain or Application Engineering projects, independent technical experts, and upper technical management.

Catalog Assets

The goal of the Catalog Asset processes is to incorporate accepted assets into a library to make them accessible to library users. The Catalog Assets IDEF₀ diagram is shown in Figure 63. Catalog Assets includes the Classify Assets, Describe Assets, Install Assets, and Review Cataloged Assets processes.

- **Classify Assets** processes determine where an asset belongs within the classification schemes in the library data models created by the Develop Library Data Model processes.
- **Describe Assets** processes create, capture, or adapt all the information that is needed to describe the asset in the context of the library's data model, once the asset has been classified. Asset description might also involve identifying dependencies on and relationships to other assets.
- **Install Assets** processes install the classified and described asset in the library system. This involves capturing the asset and its descriptive information in some kind of data base or other persistent store, and may also involve bringing the asset under configuration management control and performing other environment-specific operations.

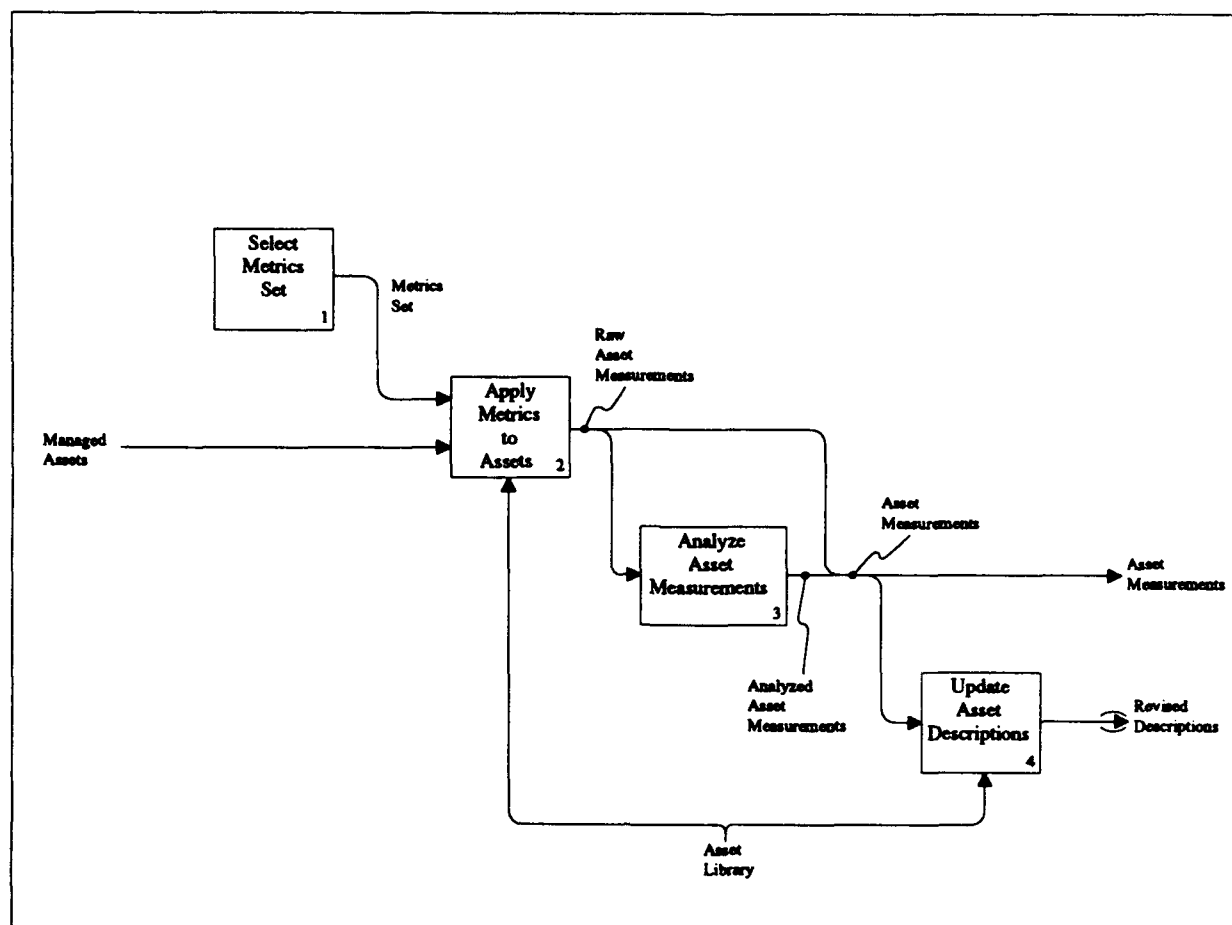
Figure 63: Catalog Assets IDEF₀ Diagram

- **Review Cataloged Assets** processes reviews the asset classification, description, and installation to ensure accuracy and quality.

Collect Asset Metrics

The goal of the Collect Asset Metrics processes is to collect information that can be used to assess the characteristics and effectiveness of individual assets within a library. Asset metrics can include measures such as asset size, asset complexity, frequency of retrieval of a particular asset, and number of problem reports associated with an asset. Such information can assist users in understanding and evaluating the assets or can be used by Asset Management staff to assess the utility and quality of particular assets. The Collect Asset Metrics processes can support the Accept Assets, Certify Assets, and Operate Library processes and can provide feedback to Domain Engineering processes.

The Collect Asset Metrics IDEF₀ diagram is shown in Figure 64. Collect Asset Metrics includes the Select Metrics Set, Apply Metrics to Assets, Analyze Asset Measurements, and

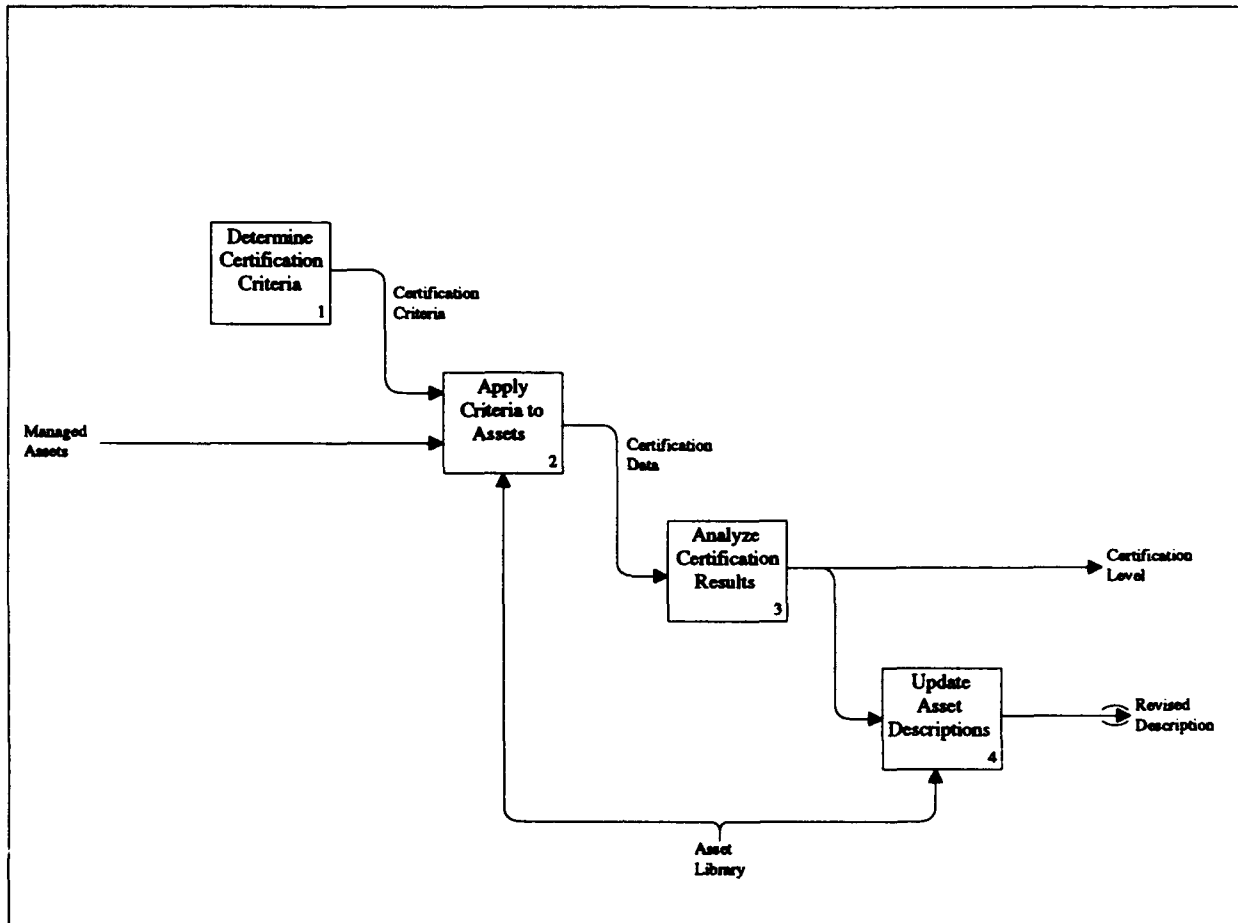
Figure 64: Collect Asset Metrics IDEF₀ Diagram

Update Asset Descriptions processes.

Certify Assets

The ultimate goal of the Certify Assets processes is to guarantee that the assets satisfy their requirements without error. Asset certification typically begins after asset acceptance and asset cataloging have occurred. However, it is typically an ongoing process that continues while an asset is available through the library system. In such cases, the certification level of the asset at any given time is usually clearly specified within the asset description.

Certification can be performed relative to arbitrary organization-defined criteria. Such criteria typically focus on asset quality or correctness as determined by inspection, testing, or formal verification. In addition to assessing general correctness, asset certification can be used to enforce or encourage organization policy by judging assets to be of higher value if they satisfy criteria that reflect the policy. An example of this is criteria that define the organization's policy on programming practices that promote reusability. A different organization may establish criteria for the same purpose, but the specifics of the criteria may be

Figure 65: Certify Assets IDEF₀ Diagram

markedly different because there is no industry consensus on what constitutes reusability.

The Certify Assets IDEF₀ diagram is shown in Figure 65. Certify Assets includes the Determine Certification Criteria, Apply Criteria to Assets, Analyze Certification Results, and Update Asset Descriptions processes.

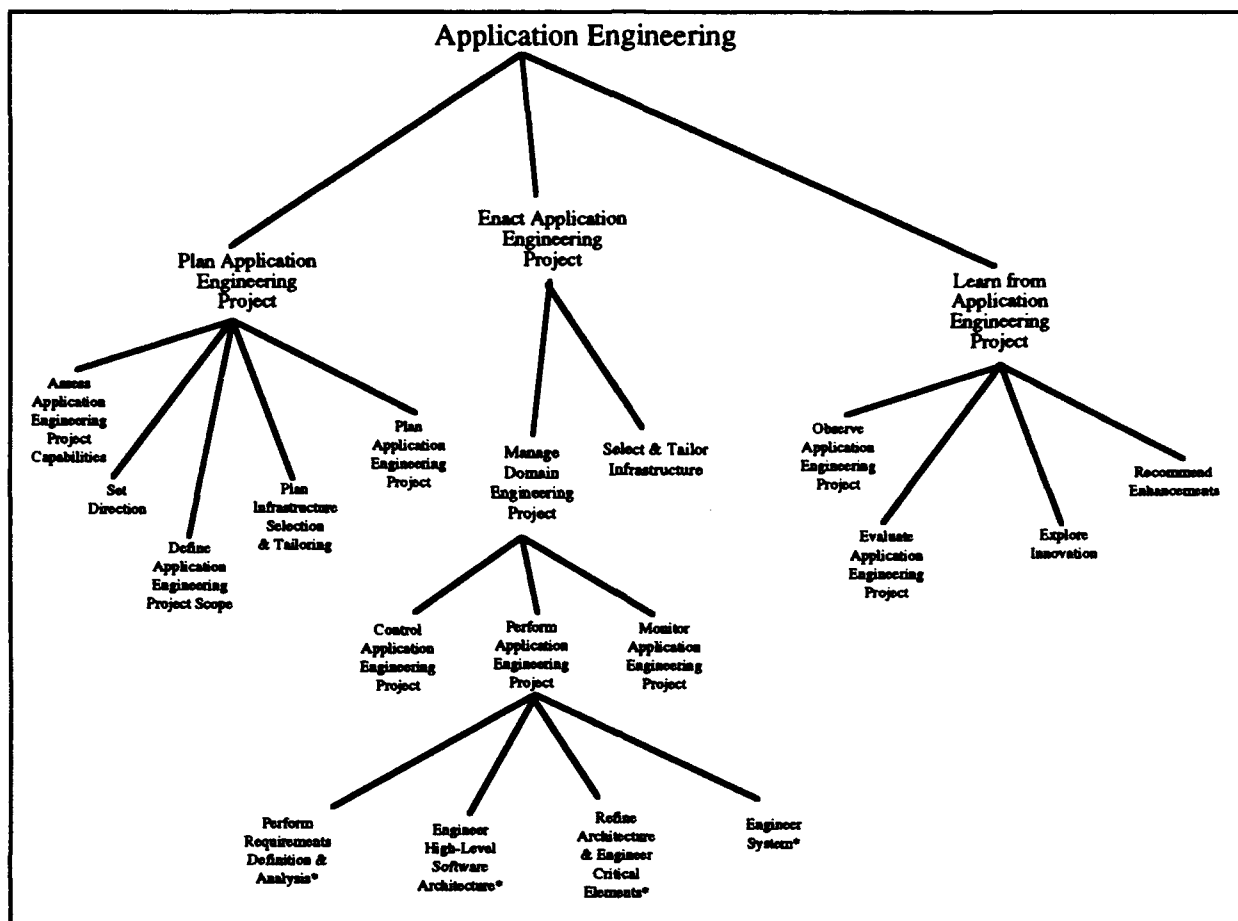


Figure 66: Application Engineering Process Abstraction Hierarchy

3.6 Application Engineering

The goal of the ROSE Application Engineering project submodel is to develop, reengineer, maintain, or evolve software systems, making use of assets created in Domain Engineering projects whenever possible. The Application Engineering submodel accommodates traditional development, and also supports reuse or reverse engineering from existing systems. Reuse of assets made available by the Asset Management processes is accomplished in Application Engineering through identifying, selecting, and tailoring desired assets and integrating them to construct an application system encompassing the target domain(s). The Application Engineering activities are derived from a spiral life cycle model and MIL-STD-2167A [15] documentation, but other life cycle models and documentation can be used by tailoring the Application Engineering activities as necessary. The process abstraction hierarchy diagram for Application Engineering is shown in Figure 66.

The ROSE PM promotes the viewpoint that both reengineering and new development should be addressed principally from a Domain Engineering perspective. In this view, software engineering focuses on model-based analyses that emphasize creation, evolution, and reuse of adaptable software architectures and components/generators (i.e., assets) from which

application systems are constructed. Software evolution activities focus on evolving the assets rather than the individual legacy systems. In a single-system context, the basis of Domain Engineering is the family of systems represented by the past, present, and future versions of the system.

However, the ROSE Application Engineering submodel accommodates more conventional reengineering and new development as needed, because Domain Engineering is not always justifiable economically or for other reasons.

Figure 67 and 68 shows a table of the activities within Application Engineering, broken into two parts. The activities are broken down into Plan, Enact, and Learn process families. Project Performance is part of Enact, so the Project Performance process categories are embedded in the Enact process family in the table. The Application Engineering Project Performance process categories include Perform Requirements Definition and Analysis, Engineer High-Level Software Architecture, Refine Architecture and Engineer Critical Elements, and Engineer System.

Another view of the Application Engineering activities, showing the flow of information among them, can be seen in the IDEF₀ diagrams. Figure 69 shows the top-level IDEF₀ Diagram for Application Engineering. This diagram shows that the **Application Engineering Process** is broken hierarchically into the **Plan Application Engineering Project**, **Enact Application Engineering Project**, and **Learn from Application Engineering Project Processes**.

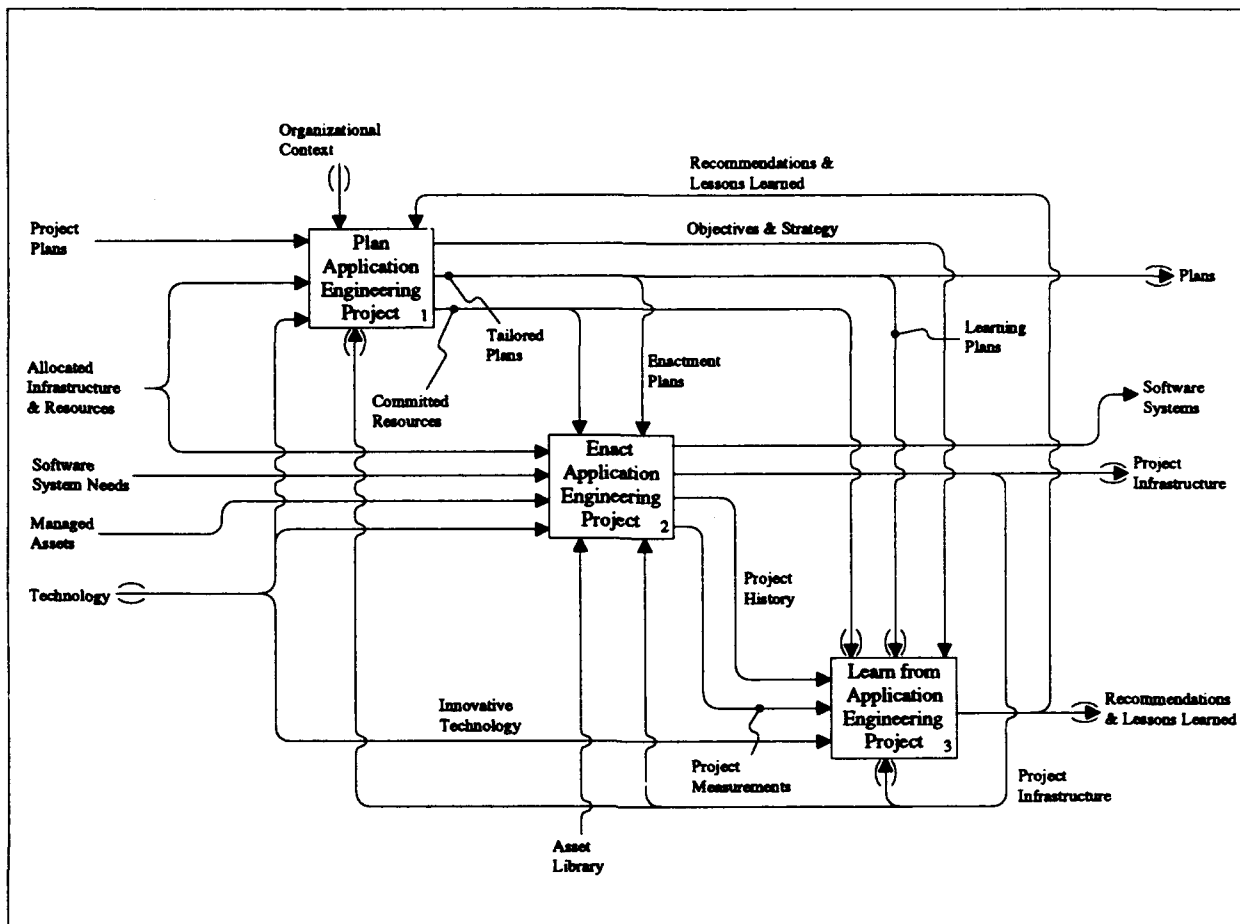
Application Engineering inputs include **Managed Assets** used to construct solution systems in the domain; **Software System Needs** that identify the overall application system requirements (possibly including internal reuse-related requirements pertaining to specific asset characteristics); **Project Plans** inherited from organizational program planning; **Allocated Infrastructure and Resources** that can be tailored to the project; and **Technology** that can contribute to the project's infrastructure and can be applied to automate processes. Controls on Application Engineering include **Organizational Context** that guides and constrains various aspects of the project to ensure that they are consistent with overall organization strategies, policies, etc. Mechanisms used by Application Engineering include the **Asset Library** that provides the organizing scheme for a set of assets and capabilities for accessing and processing the assets. Outputs from Application Engineering include project **Plans**, the tailored **Project Infrastructure**, **Recommendations and Lessons Learned** from project management; and **Software Systems** that are constructed from assets and other application products.

The following subsections describe the activities carried out in an Application Engineering Project. These are divided into the Plan-Enact-Learn project management activities described briefly in Section 3.6.1, and the project engineering activities (performed within the Enact family) described in Section 3.6.2.

Figure 67: Application Engineering Table - Part 1

Category	Plan	Enact	Learn
3 Refine Architecture & Engineer Critical Elements	Refine Project Plans Assess Progress Revise Plans Revise Resource Allocation Revise Project Schedule Resolve Issues as Required Refine Reuse Plan Develop Integration Test Plan Conduct Program Mgmt. Reviews (PMRs) Assess Risks	Engineer Software Architecture Select & Tailor Architecture Assets Reverse Engineer Architecture Develop Remaining Architecture Integrate Architecture Develop User Interface Document Engineer Critical Element Prototypes Select & Tailor Critical Element Assets Generate Critical Elements Scavenge Critical Elements Develop Remaining Critical Elements Integrate Critical Element Prototypes Assess Performance of Critical Elements Perform Preliminary Design Select & Develop Preliminary Design Select & Tailor Prelim. Design Assets Reverse Engineer Preliminary Design Develop Remaining Preliminary Design Integrate Preliminary Design Develop Design Documentation SDD/IDD, SDFs Perform COTS/NDI Evaluation Determine Language Requirements Document Engineering Notes	Peer Reviews / Walkthroughs Conduct PDR
4 Engineer System	Refine Project Plans Assess Progress Revise Resource Allocation Revise Project Schedule Resolve Issues as Required Develop Programming Standards Conduct Program Mgmt. Reviews (PMRs) Assess Risks	Perform Detailed Design Select & Develop Detailed Design Select & Tailor Detailed Design Assets Reverse Engineer Detailed Design Develop Remaining Detailed Design Integrate Detailed Design Develop Implementation Prototypes Develop Event-driven prototypes Engineer Software Components Select & Develop Components Select & Tailor Sw. Component Assets Generate Software Components Scavenge Software Components Develop Remaining Sw. Components Develop Documentation Reuse Guidelines Engineering Notes Documentation of Reusable Assets User Documentation Doc. of Maintainability & Evolvability Perform Maintenance Training Develop Operation Plan	Conduct Detailed Design Reviews & Walkthroughs Conduct CDR Develop Component Test Plan Conduct Peer Reviews & Walkthroughs Perform Testing & Integration Perform Component Test Perform Software integration & Test Conduct Test Readiness Review (TRR) Perform System Test Test system performance Perform FQT, FCA, PCA
Learn	Observe Application Engineering Project	Evaluate Application Eng. Project	Explore Innovation Recommend Enhancements

Figure 68: Application Engineering Table - Part 2

Figure 69: Application Engineering IDEF₀ Diagram

3.6.1 Manage Application Engineering

This section describes the processes carried out in managing an Application Engineering project. The processes are described briefly since they are generally the same set of processes described in Organization Management, except with a project focus. The differences between Application Engineering project management and Organization Management are stressed in this section.

3.6.1.1 Plan Application Engineering Project

Figure 70 contains an IDEF₀ Diagram for the Plan Application Engineering Project process family. As shown in the figure, the Plan process consists of the following five process categories:

- **Assess Application Engineering Project Capabilities** processes characterize the current state of Application Engineering practice and capabilities within the organiza-

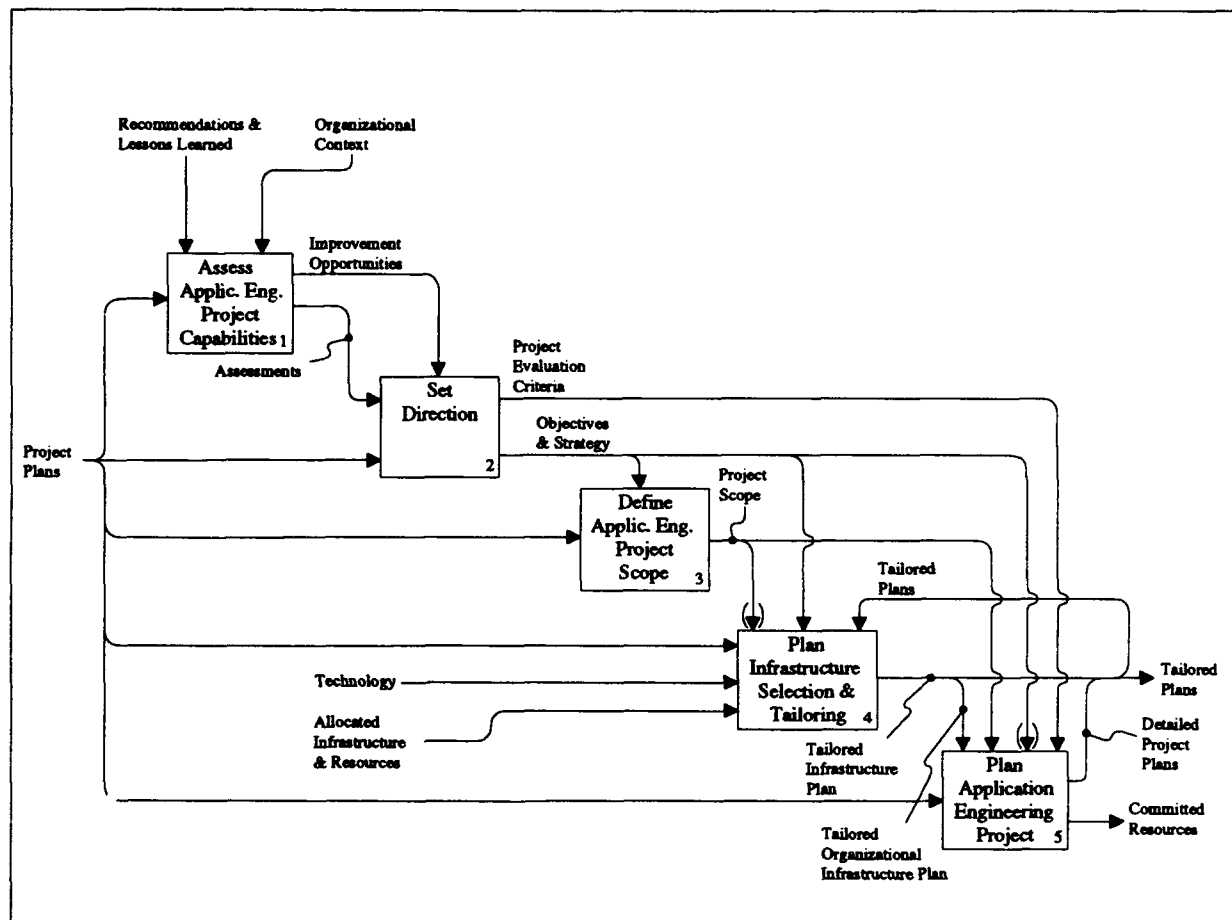


Figure 70: Plan Application Engineering Project IDEF₀ Diagram

tion. The assessment considers a variety of factors, including application engineering and asset utilization experience, domain expertise, available support technology, and availability and quality of legacy systems and assets. Included in this process is a review of recommendations and lessons learned from previous cycles to identify specific improvements to be made to the organization's Application Engineering capabilities.

- **Set Direction** processes determine a strategy and objectives for the Application Engineering project. A high-level strategy may be defined in the Plan Projects process category within Organization Management. In Set Direction, a more detailed strategy and objectives are defined, based on the high-level strategy. Once a strategy and objectives have been developed, success criteria are identified to determine whether the objectives are met.
- **Define Application Engineering Project Scope** processes bound the scope of key aspects of the project. This may include defining the scope of the domain(s) from which assets will be drawn to produce the application, in more detail than was done in the Scope Line of Business process in Organization Management. It may also involve constraining a variety of other aspects of the project, such as acceptable levels

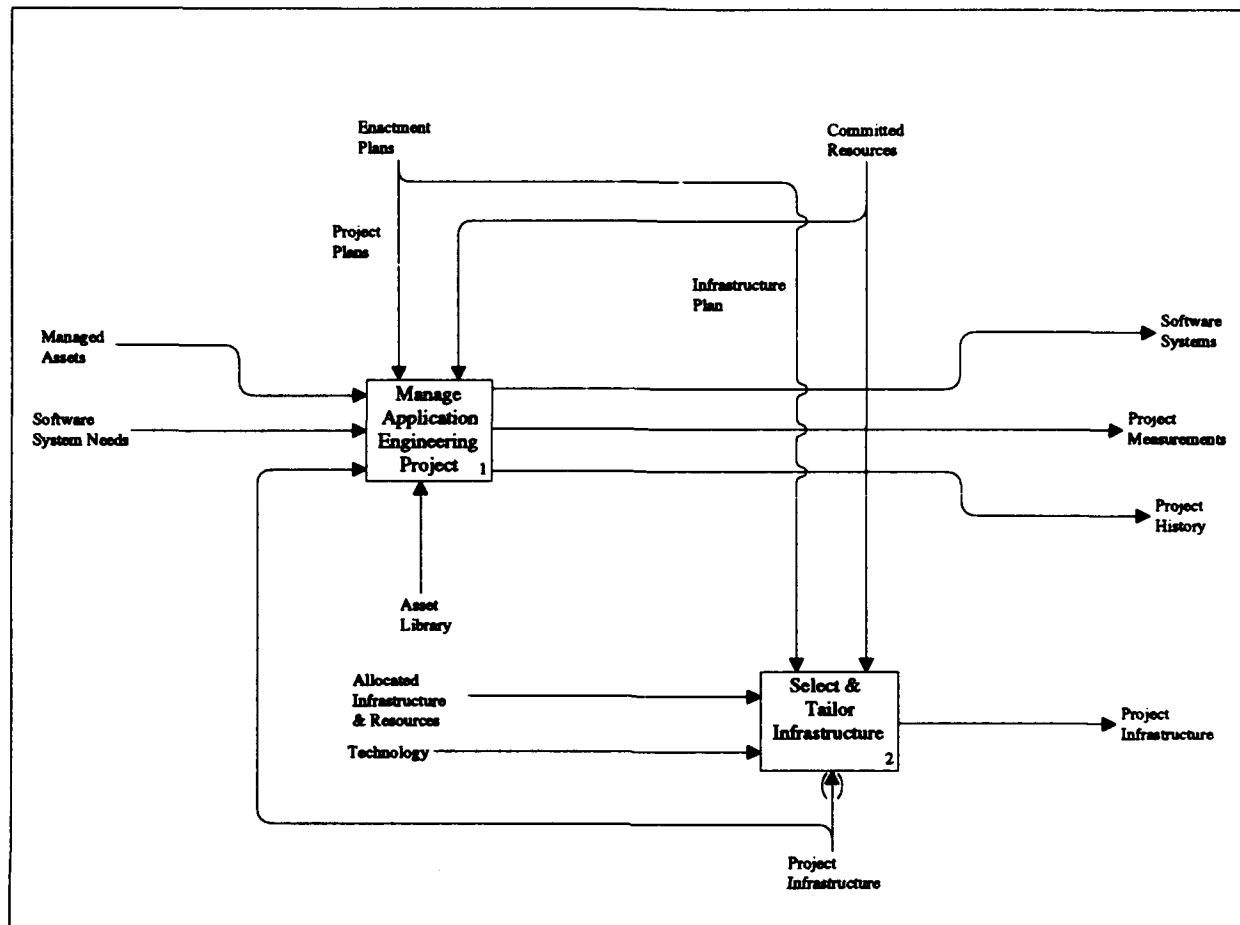


Figure 71: Enact Application Engineering Project IDEF₀ Diagram

of application functionality and quality, although many of these issues may be left in whole or in part to the Perform Requirements Definition and Analysis processes.

- **Plan Infrastructure Selection and Tailoring** processes plan the technical, organizational, and educational infrastructure needs of the project. These needs can be satisfied by the infrastructure at the organizational level, tailored from the infrastructure at the organizational level, or, if necessary, created specifically for the project. Project infrastructure planning includes plans for process, policies, standards, and training.
- **The Plan Application Engineering Project** processes perform detailed project planning. Detailed project planning includes the development of a schedule, budget, and resource needs based on the high-level schedule and budget developed in the Plan Projects process category in Organization Management. Political, technical, schedule, and cost constraints on the project are identified and control mechanisms are planned. Project risks are identified and evaluated as “high”, “medium”, or “low”. Risk mitigation plans and mechanisms are developed for handling high risks.

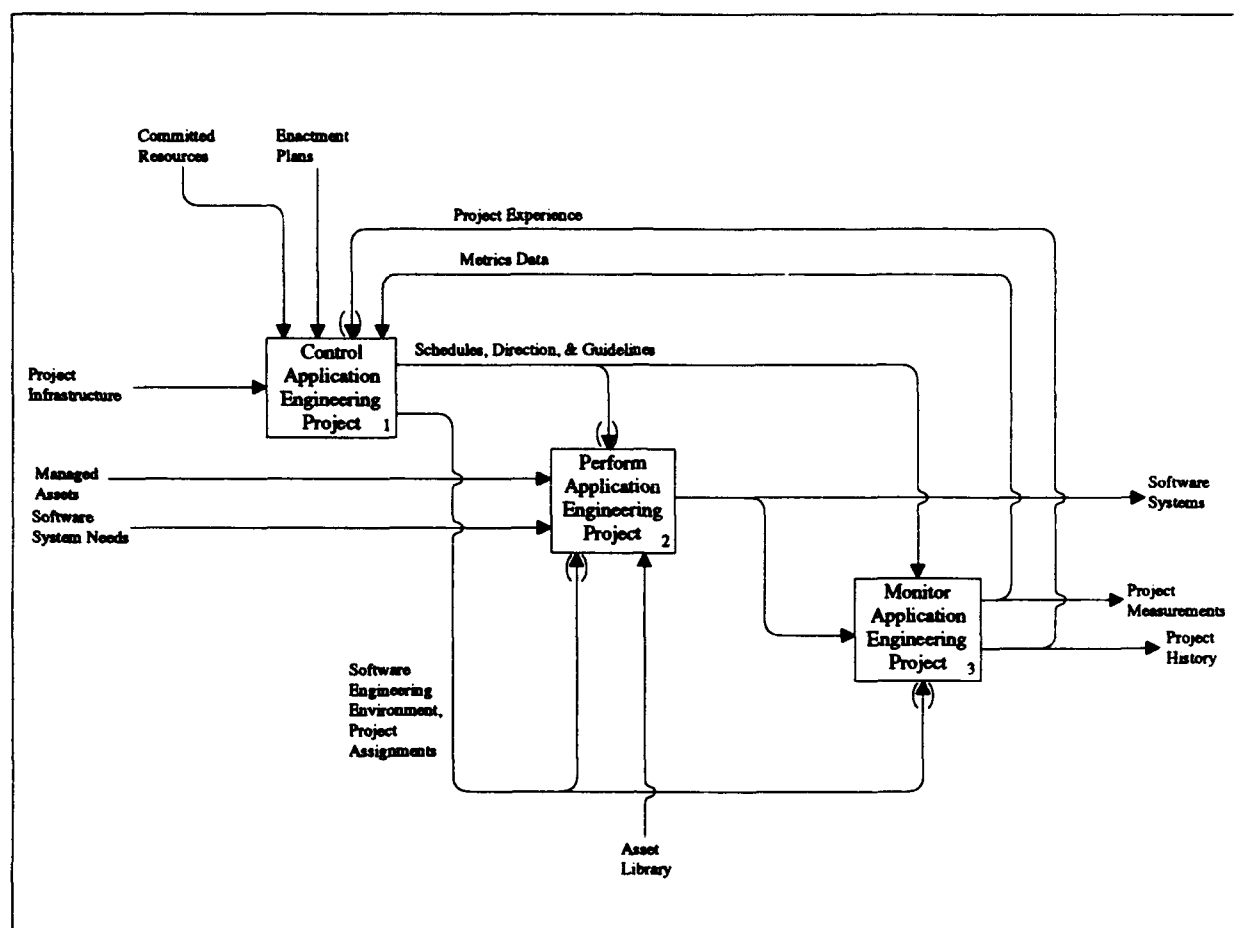


Figure 72: Manage Application Engineering Project IDEF₀ Diagram

3.6.1.2 Enact Application Engineering Project

The Enact Application Engineering Project process family manages the day-to-day activity of the project and ensures that an infrastructure sufficient to meet the needs of the project is established and maintained.

Figure 71 contains an IDEF₀ diagram for the Enact Application Engineering Project process family. As shown in the figure, the Enact process family consists of the **Manage Application Engineering Project** and **Select and Tailor Infrastructure** process categories.

Manage Application Engineering Project

The Manage Application Engineering Project process category establishes a temporal context for the performance of the Application Engineering project and performs detailed project supervision. The Manage Application Engineering Project IDEF₀ diagram is shown in Figure 72. The Manage Application Engineering Project process includes Control Application Engineering Project, Perform Application Engineering Project, and Monitor Application

Engineering Project.

The **Control Application Engineering Project** process activities intervene with project performance in order to keep the project on track relative to objectives set during Organizational Planning. Project control is the "management" function (in the most conventional sense) for the Application Engineering project. The goal of project control activities is to optimize overall project performance, as measured by factors such as the degree to which assets are utilized in constructing applications, the overall quality of the applications, the effectiveness of the infrastructure, etc.

The **Perform Application Engineering Project** activities at the core of Enactment are where the engineering processes being enacted are "hooked in" to Application Engineering project management and actually performed by individual staff members. These engineering processes are described in Section 3.6.2. From a management perspective, project performance activities also involve tactical decisions about *how* the work will be performed on a detailed, day-to-day basis. At one level, project performance activities can be viewed as individualized techniques for filling in the minute implementation details that are generally missing from project processes established in Project Planning.

The **Monitor Application Engineering Project** activities capture "learning-oriented" information from the Application Engineering project as it is performed. Some of this information provides feedback to project control activities in order to adjust schedules, budgets, milestones, incentives, guidelines, or task assignments. Some of the same information may serve as input to the Learn from Application Engineering Project processes, along with data collected in accordance with process and product metrics identified in Plan Application Engineering Project, and project history data such as rationale for decisions made and interim work products created.

Select and Tailor Infrastructure

Select and Tailor Infrastructure processes select the infrastructure to meet the project's needs from the organization's infrastructure. This infrastructure may be tailored to the project and missing infrastructure capabilities to satisfy new project-specific needs may be developed.

3.6.1.3 Learn from Application Engineering Project

Processes in the Learn from Application Engineering Project process family assess the overall success of the strategies and plans put into place by the Plan Application Engineering Project process family, comparing project results and experience against project objectives, to improve the quality of the plans and infrastructure. Figure 73 contains an IDEF₀ diagram for the Learn from Application Engineering Project process family. As shown in the figure, the Learn process family consists of **Observe Application Engineering Project**, **Evaluate Application Engineering Project**, **Explore Innovation**, and **Recommend Enhancements**. These process categories were discussed in section 3.3.3, on Organization

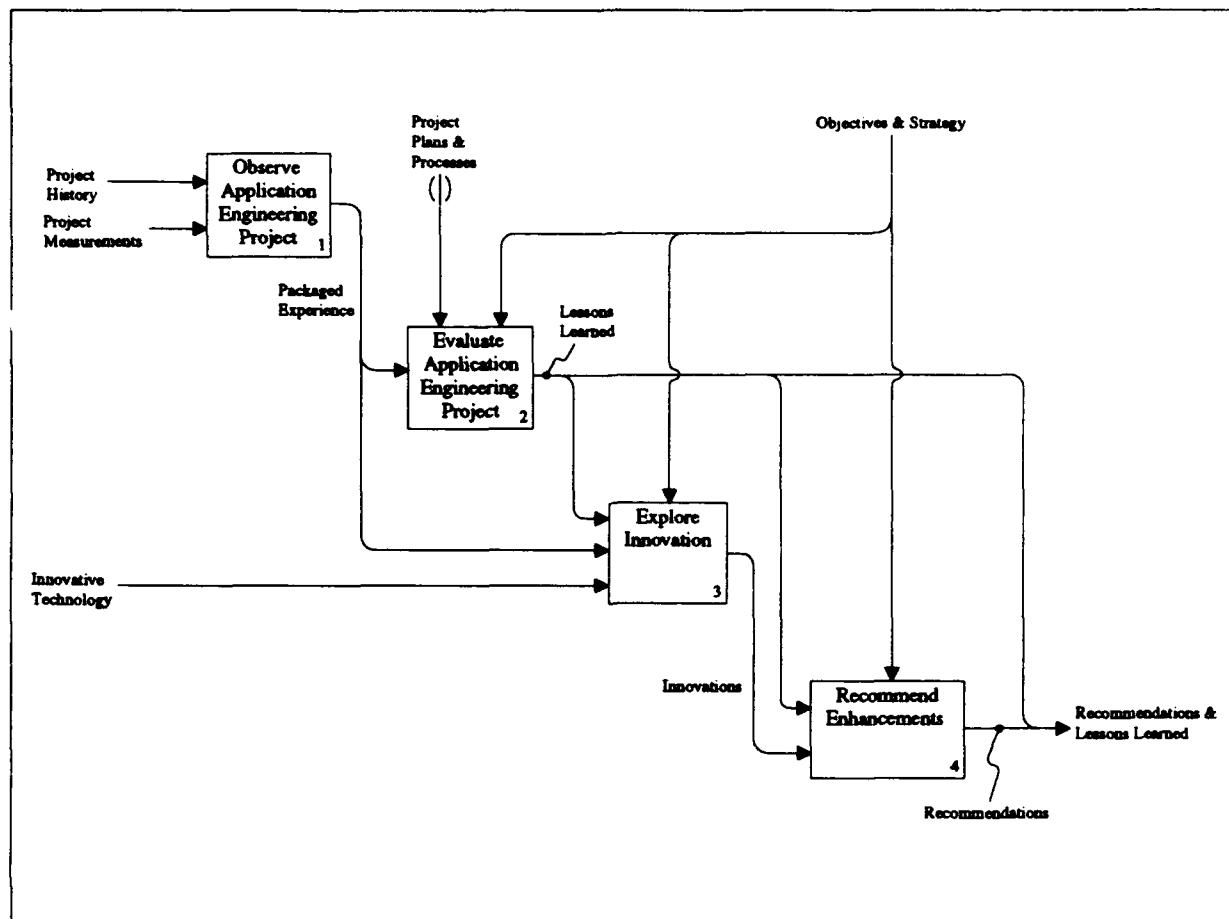


Figure 73: Learn from Application Engineering Project IDEF₀ Diagram

Management Learning. Project level learning is similar to organizational learning, except that the focus is on the individual Application Engineering project.

3.6.2 Perform Application Engineering

The **Perform Application Engineering Project** process included among the Enact Application Engineering Project processes described above is where the engineering activities enacted in the project are “hooked in” to project management activities. This section describes these engineering activities, divided into process categories that include their own localized Plan-Enact-Learn activities.

A repeated theme within the Application Engineering processes is the following sequence of activities:

- *Select and tailor assets* to produce system work products
- *Reverse engineer existing system artifacts* to produce work products where no appli-

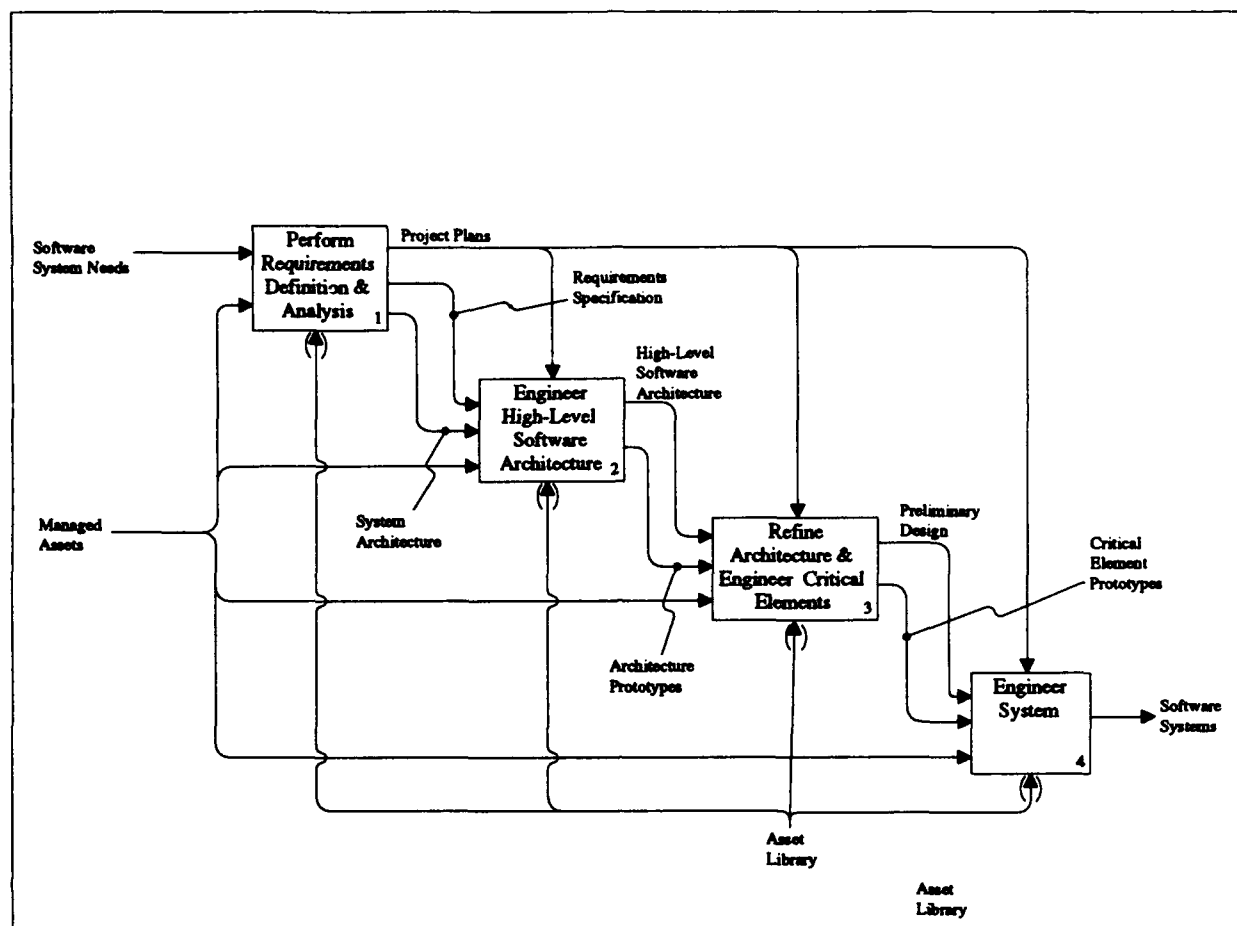


Figure 74: Perform Application Engineering Project IDEF₀ Diagram

cable assets exist

- *Develop new work products* where no applicable assets or existing system artifacts exist
- *Integrate work products* produced via asset reuse, reverse engineering, and new development to produce integrated system elements

This key aspect of the ROSE PM reflects the general approach taken to integrating CFRP Asset Utilization processes within Application Engineering.

The Perform Application Engineering Project IDEF₀ diagram is shown in Figure 74. Perform Application Engineering Project includes Perform Requirements Definition and Analysis, Engineer High-Level Software Architecture, Refine Architecture and Engineer Critical Elements, and Engineer System. Each of these process categories is described in detail below.

3.6.2.1 Perform Requirements Definition and Analysis

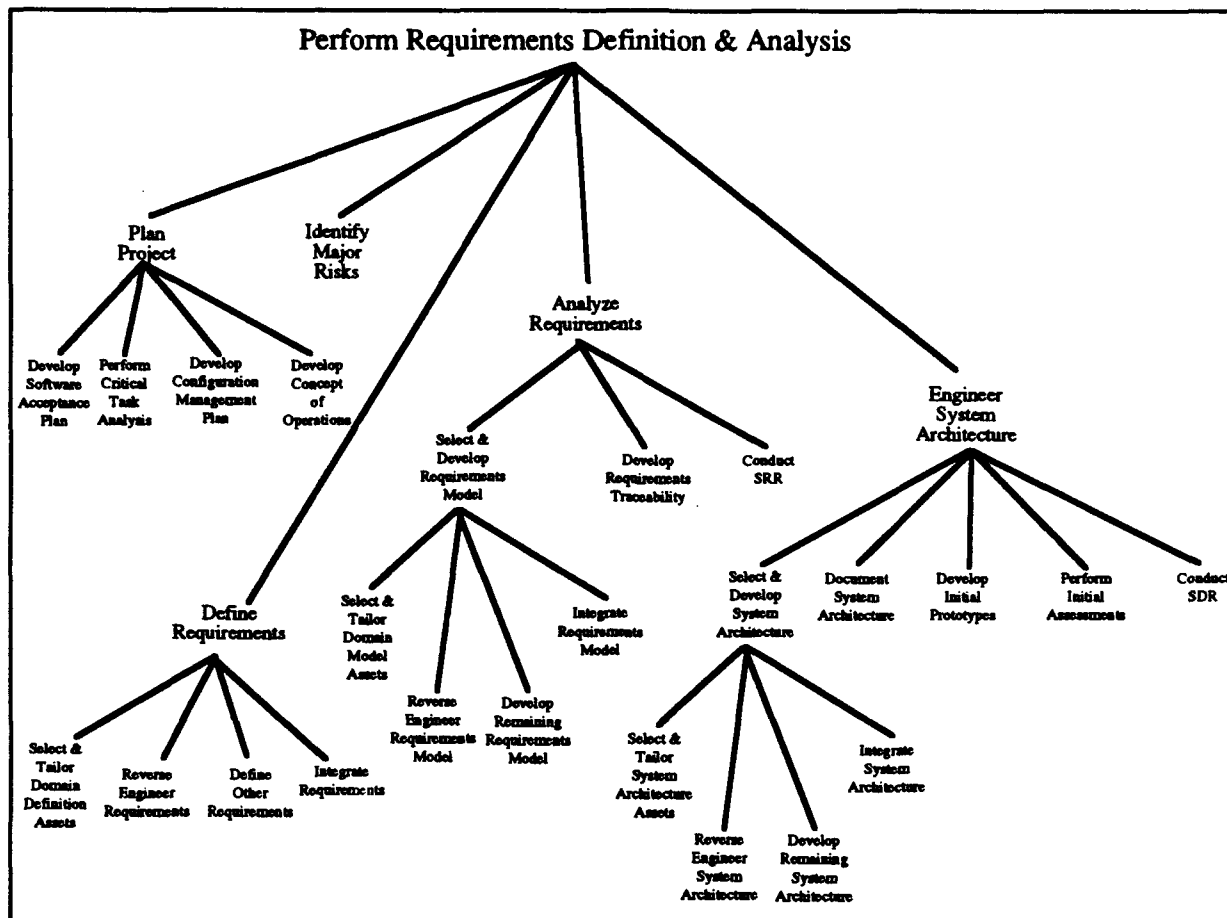


Figure 75: Perform Requirements Definition and Analysis Process Abstraction Hierarchy

Perform Requirements Definition and Analysis processes define and analyze the requirements for the application system and engineer the high-level system architecture. These processes also include preliminary detailed planning of technical functions. The process abstraction hierarchy diagram for Perform Requirements Definition and Analysis is shown in Figure 75.

The Perform Requirements Definition and Analysis IDEF₀ diagram is shown in Figure 76. Perform Requirements Definition and Analysis includes the Plan Project, Identify Major Risks, Define Requirements, Analyze Requirements, and Engineer System Architecture processes.

3.6.2.1.1 Plan Project

Plan Project processes support detailed project planning. The Plan Project IDEF₀ diagram is shown in Figure 77. Plan Project includes the Develop Software Acceptance Plan, Perform Critical Task Analysis, Develop Configuration Management Plan, and Develop Concept of Operations processes. These processes may refine the objectives and the scoping or process decisions inherited from the overall project planning processes.

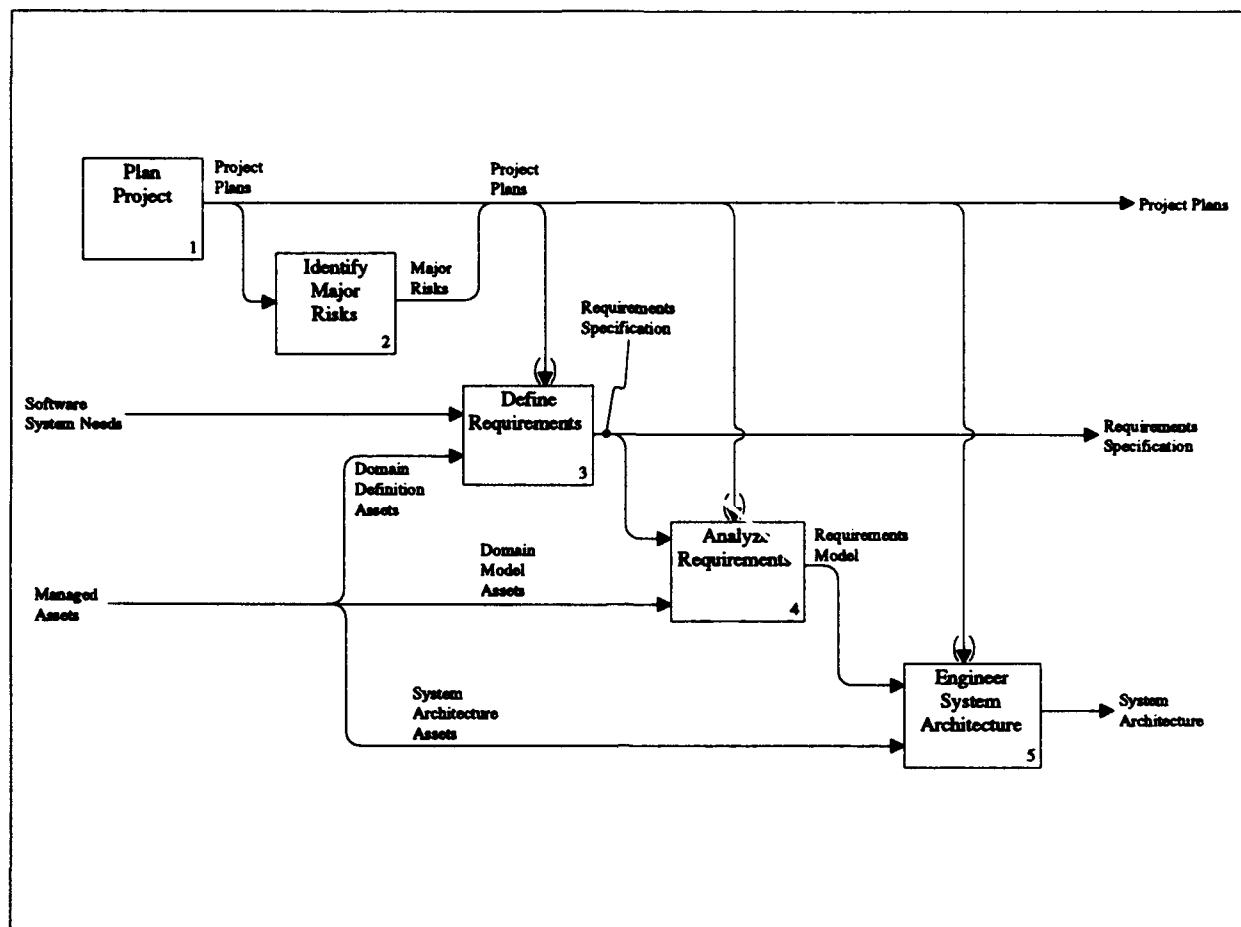


Figure 76: Perform Requirements Definition and Analysis IDEF₀ Diagram

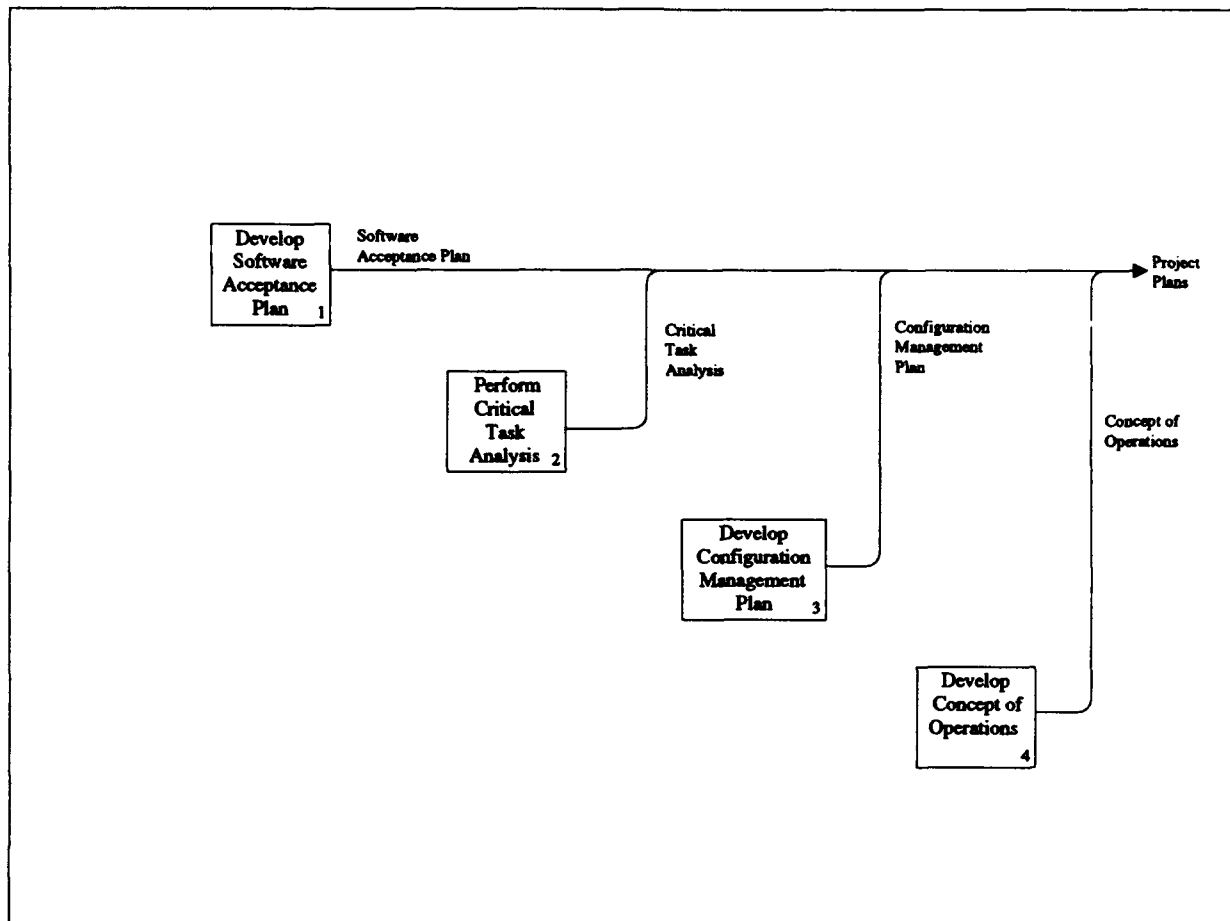
3.6.2.1.2 Identify Major Risks

Identify Major Risks processes identify and analyze major project risks and identify potential risk mitigation strategies. It is very important to identify high risk areas as early as possible on the project so that an effective strategy for resolving these issues can be determined.

3.6.2.1.3 Define Requirements

Define Requirements processes perform requirements definition activities. On many projects the requirements may be already defined by the customer. In this case, interpretation and elaboration of the requirements is needed to ensure their understanding, and it is this interpretation and elaboration of requirements that is produced here.

The Define Requirements IDEF₀ diagram is shown in Figure 78. Define Requirements includes the Select and Tailor Domain Definition Assets, Reverse Engineer Requirements, Define Other Requirements, and Integrate Requirements processes.

Figure 77: Plan Project IDEF₀ Diagram

- **Select and Tailor Domain Definition Assets** processes look for assets that define, scope, or bound the domain, or otherwise express requirements on systems within the domain, which can be reused to define system requirements. Possible candidate assets are identified and then selected through an evaluation process. These assets may need tailoring to customize them to the project.
- **Reverse Engineer Requirements** processes look to existing systems related to the system under development to reverse engineer requirements definitions that could not be obtained from the asset base. Again, tailoring of the requirements definitions may be necessary.
- **Define Other Requirements** processes define from scratch any remaining requirements that could not be obtained through selecting and tailoring assets or reverse engineering.
- **Integrate Requirements** processes integrate the requirements definitions together, including requirements defined by selecting and tailoring domain definition assets, requirements defined through reverse engineering, and requirements defined from scratch.

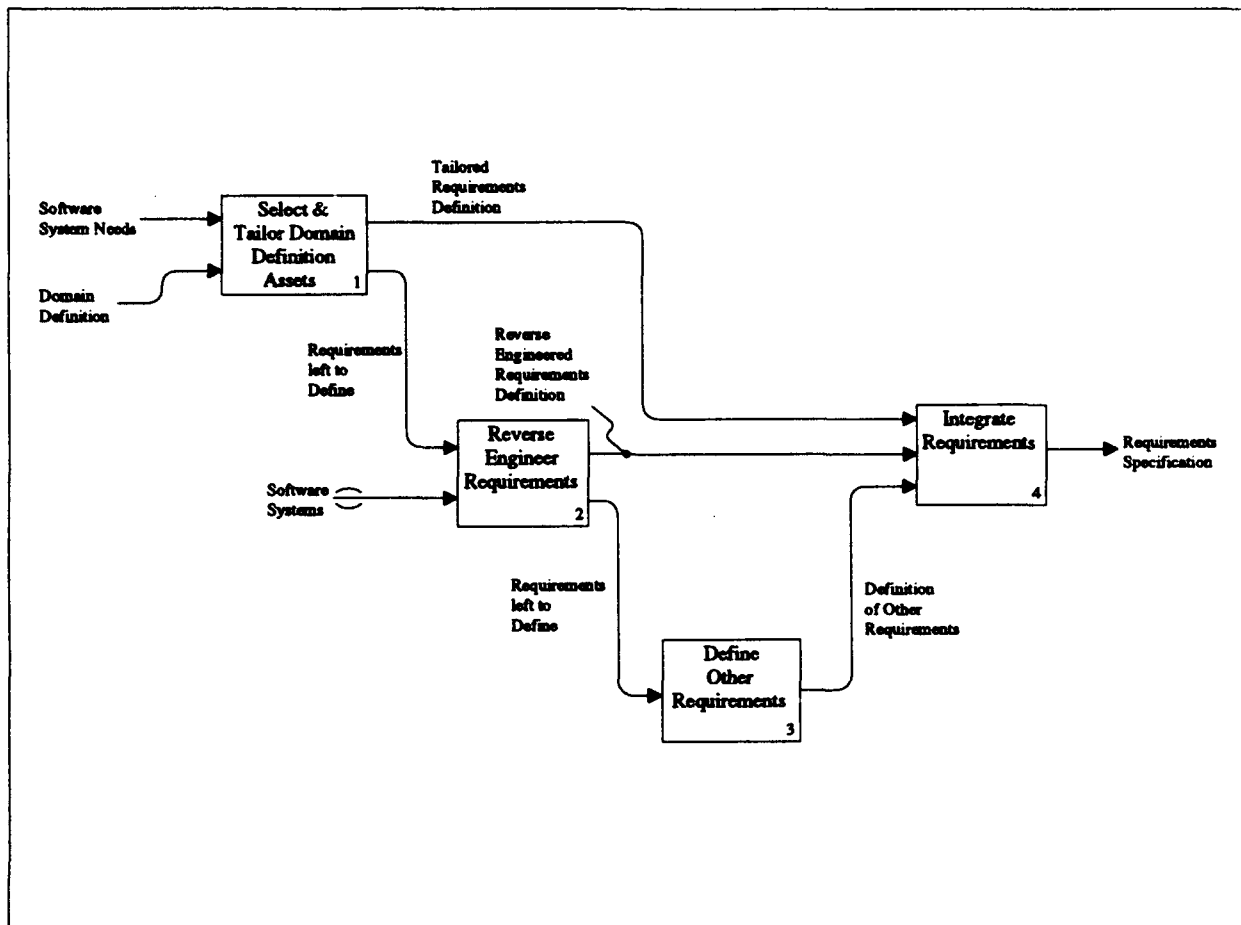


Figure 78: Define Requirements IDEF₀ Diagram

Some tailoring may be needed for the requirements to fit together well.

3.6.2.1.4 Analyze Requirements

Analyze Requirements processes model the system requirements to provide a more formal basis for architecture development and design. The processes also develop a method to trace requirements throughout the Application Engineering process.

The Analyze Requirements IDEF₀ diagram is shown in Figure 79. Analyze Requirements includes the Select and Develop Requirements Model, Develop Requirements Traceability, and Conduct SRR processes.

Select and Develop Requirements Model

Select and Develop Requirements Model processes develop the requirements model through reuse, reverse engineering, and/or new development.

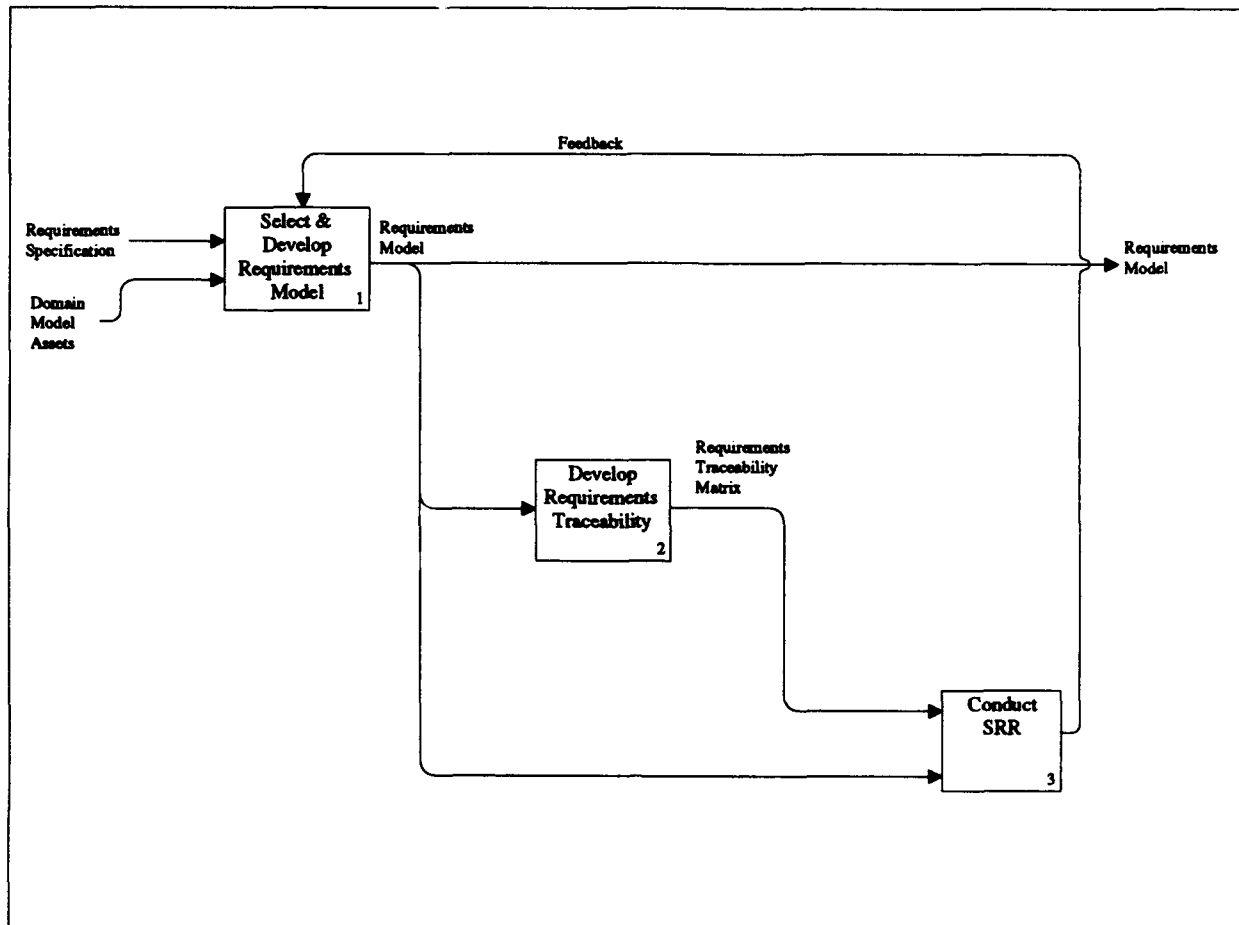


Figure 79: Analyze Requirements IDEF₀ Diagram

The Select and Develop Requirements Model IDEF₀ diagram is shown in Figure 80. Select and Develop Requirements Model includes the Select and Tailor Domain Model Assets, Reverse Engineer Requirements Model, Develop Remaining Requirements Model, and Integrate Requirements Model processes.

First, requirements model assets as expressed within the domain model are reused as appropriate to develop the requirements model. Reverse engineering from existing systems may be used to obtain additional portions of the requirements model that could not be obtained through reuse. As a last resort, remaining portions of the requirements model are developed from scratch. All portions are then integrated.

Develop Requirements Traceability

Develop Requirements Traceability processes develop a capability for tracing requirements throughout the design and implementation of the application system.

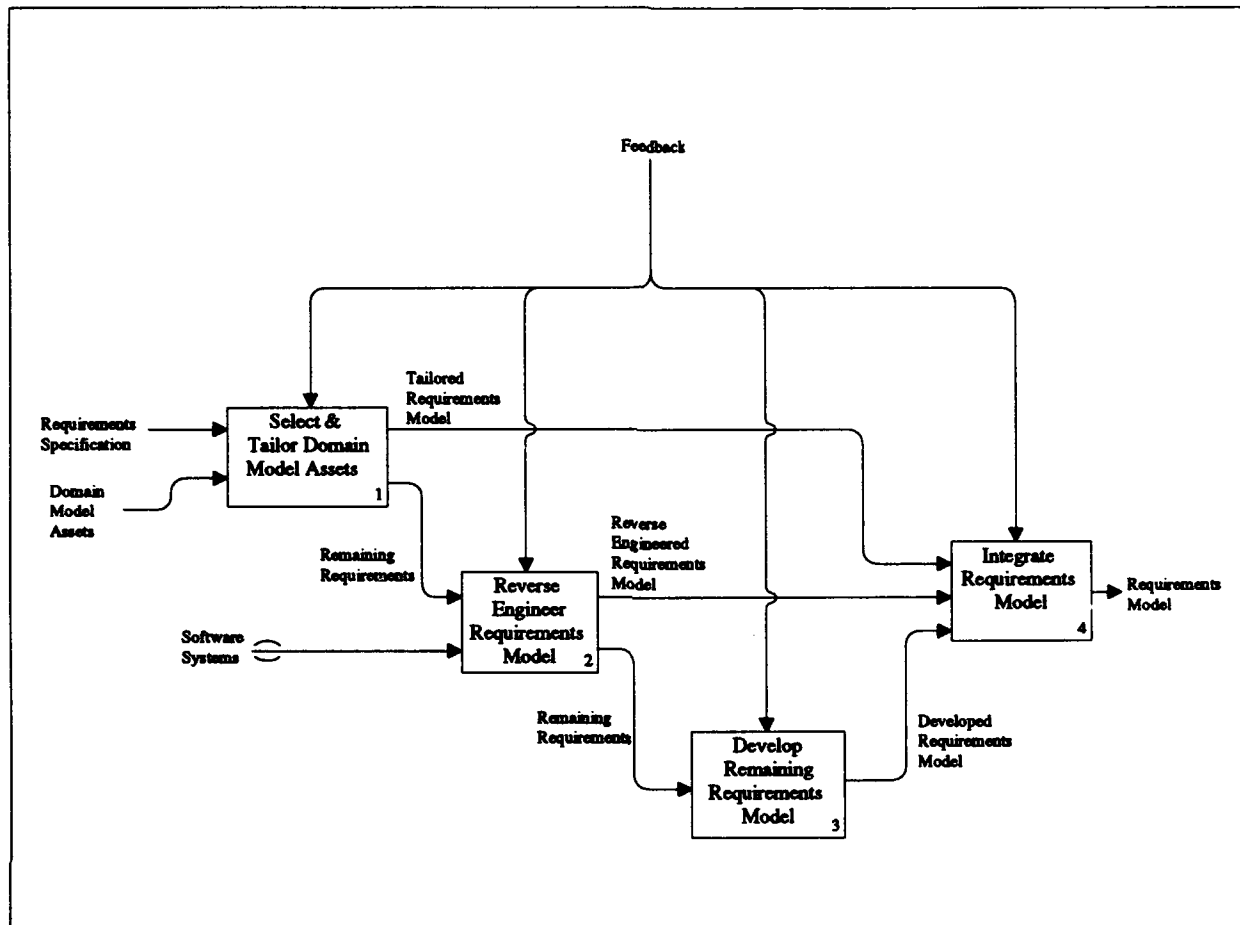


Figure 80: Select and Develop Requirements Model IDEF₀ Diagram

Conduct SRR

Conduct SRR processes involve conducting the System Requirements Review to ensure that the correct set of requirements has been obtained before the system is constructed.

3.6.2.1.5 Engineer System Architecture

Engineer System Architecture processes develop the architecture, or high-level design, of the overall system, including hardware and software. The Engineer System Architecture IDEF₀ diagram is shown in Figure 81. Engineer System Architecture includes the Select and Develop System Architecture, Document System Architecture, Perform Initial Assessments, Develop Initial Prototypes, and Conduct SDR processes.

Select and Develop System Architecture

Select and Develop System Architecture processes develop the system architecture through

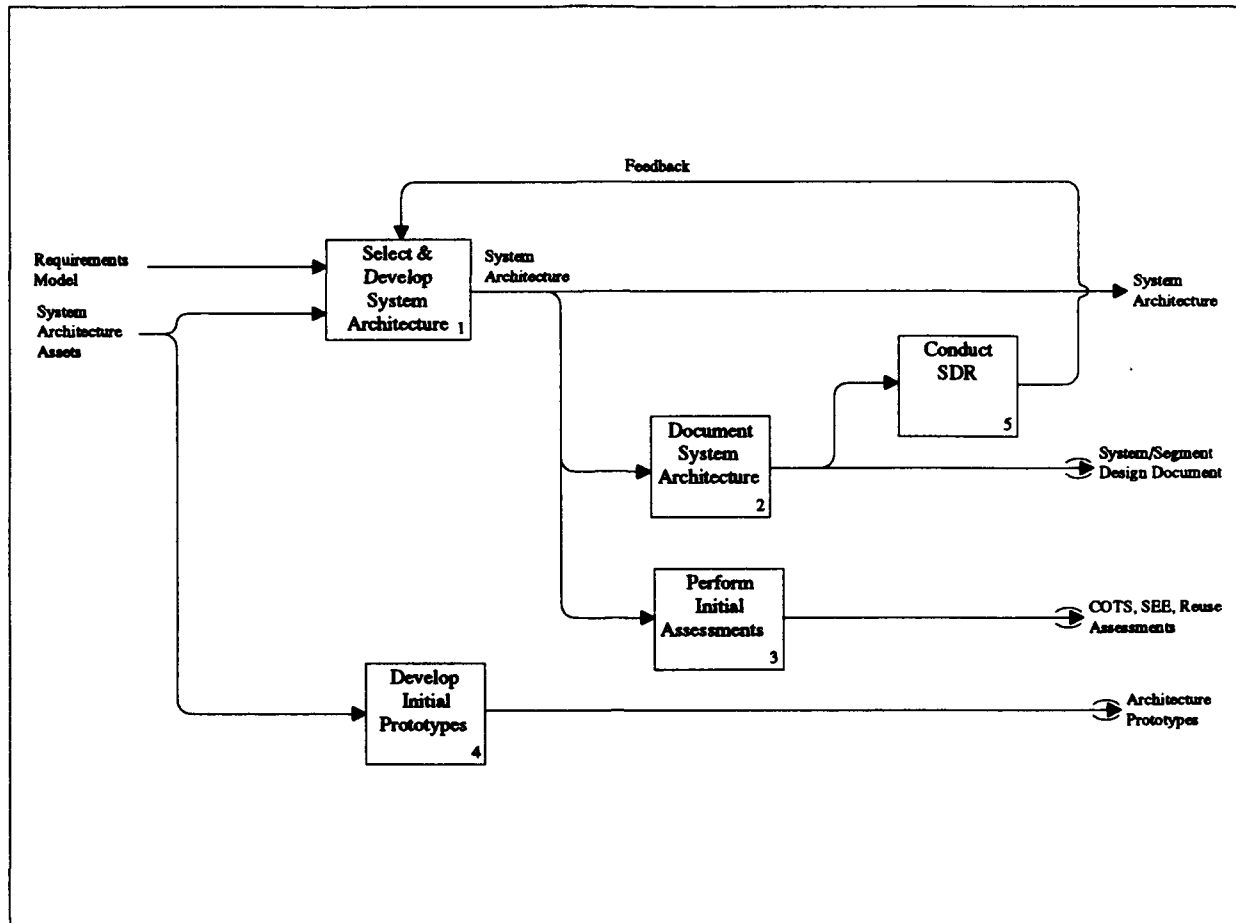


Figure 81: Engineer System Architecture IDEF₀ Diagram

reuse, reengineering, and/or new development. The Select and Develop System Architecture IDEF₀ diagram is shown in Figure 82. Select and Develop System Architecture includes the Select and Tailor System Architecture Assets, Reverse Engineer System Architecture, Develop Remaining System Architecture, and Integrate System Architecture processes.

Document System Architecture

Document System Architecture processes develop the System/Segment Specification (SSS) and System/Segment Design Document (SSDD). The SSS specifies the requirements for a system or segment of a system. The SSDD describes the design of the system/segment and its operational and support environment. It describes the organization of the system/segment as composed of Hardware Configuration Items (HWCIs), Computer Software Configuration Items (CSCIs), and manual operations.

Develop Initial Prototypes

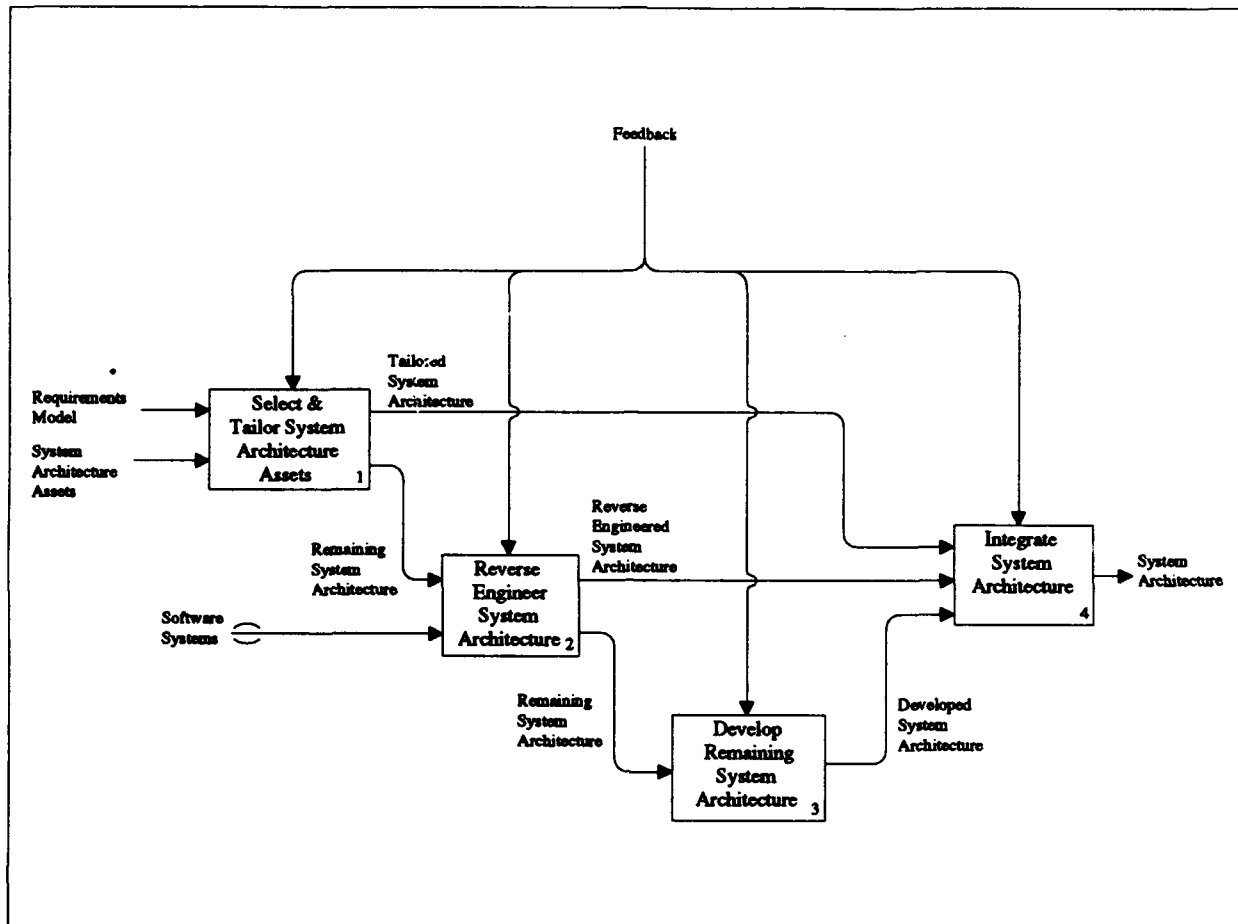


Figure 82: Select and Develop System Architecture IDEF₀ Diagram

Develop Initial Prototypes processes develop the initial system prototypes. These prototypes focus on the high risk areas identified earlier, in order to gain an understanding of these areas early in the development process.

Perform Initial Assessments

Perform Initial Assessments processes perform initial assessments of the system architecture and related issues, including an assessment of reusable assets that can be used on the system; identification of COTS products that can be used, including vendor plans for future products; and an assessment of the software engineering environment capabilities needed to carry out the project.

Conduct SDR

Conduct SDR processes conduct the System Design Review.

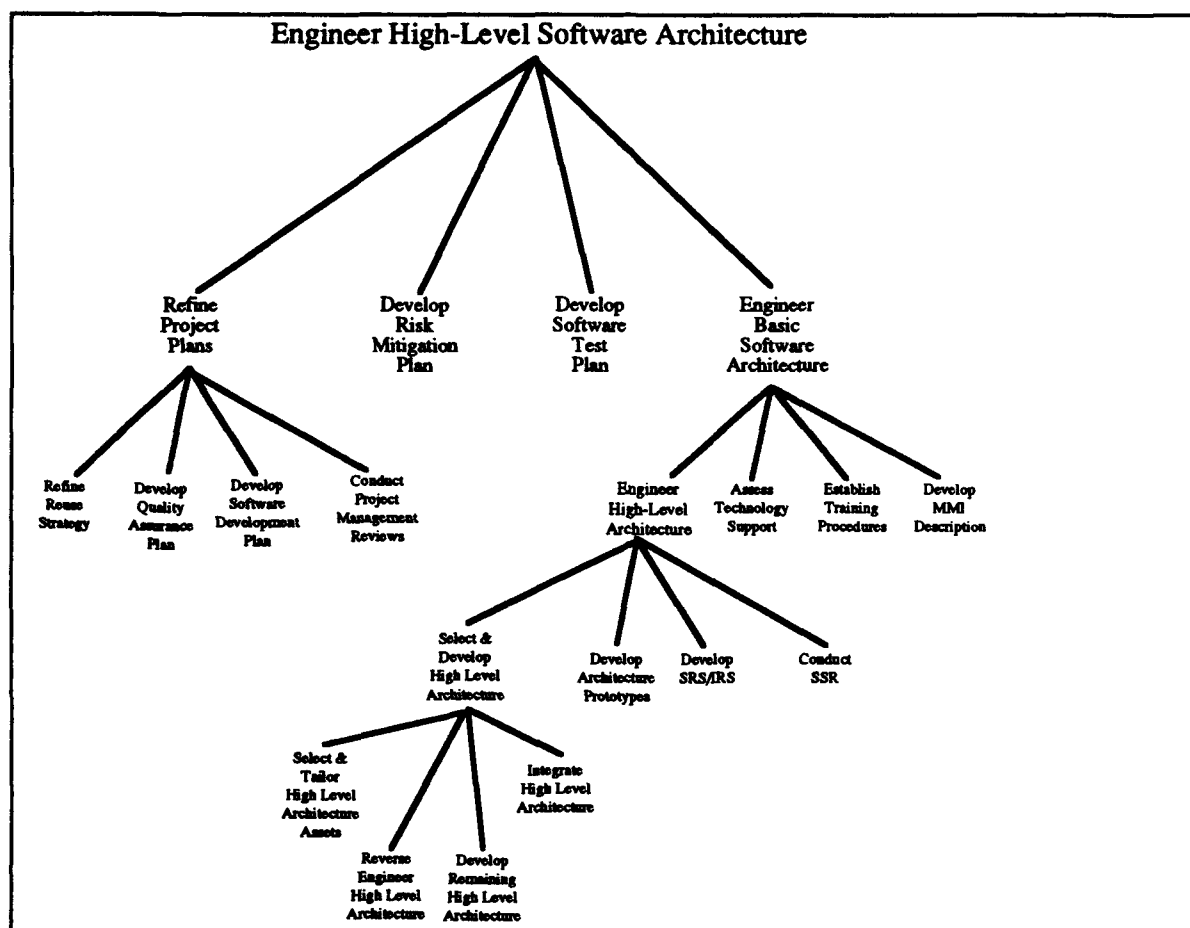


Figure 83: Engineer High-Level Software Architecture Process Abstraction Hierarchy

3.6.2.2 Engineer High-Level Software Architecture

Engineer High-Level Software Architecture processes develop the high-level architecture for the software portion of the system under development. The process abstraction hierarchy diagram for Application Engineering is shown in Figure 83.

The Engineer High-Level Software Architecture IDEF₀ diagram is shown in Figure 84. Engineer High-Level Software Architecture includes the Refine Project Plans, Develop Risk Mitigation Plan, Develop Software Test Plan, and Engineer Basic Software Architecture processes.

3.6.2.2.1 Refine Project Plans

Refine Project Plans processes refine the project plans developed during requirements definition and analysis. The Refine Project Plans IDEF₀ diagram is shown in Figure 85. Refine Project Plans includes the Refine Reuse Strategy, Develop Quality Assurance Plan, Develop Software Development Plan, and Conduct Project Management Reviews processes.

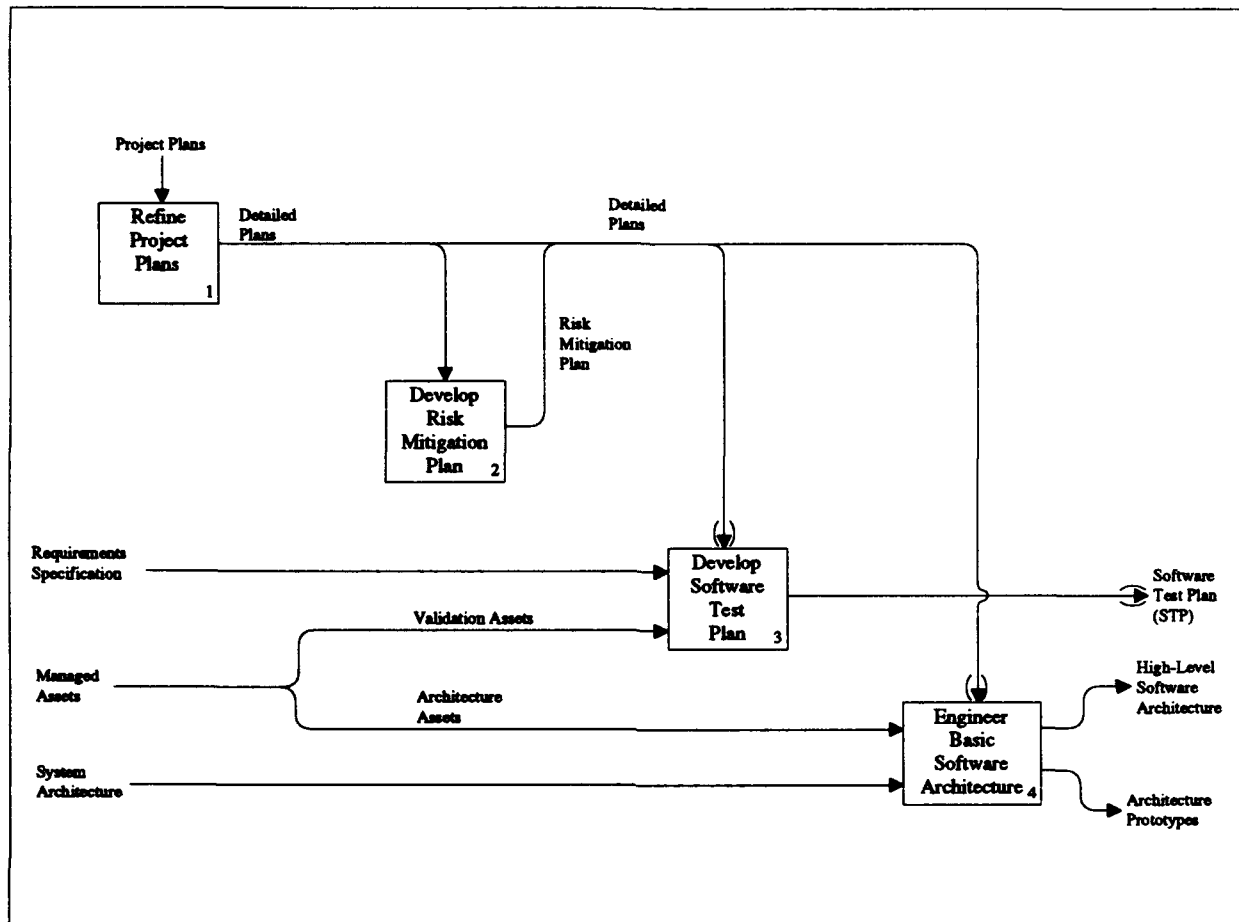


Figure 84: Engineer High-Level Software Architecture IDEF₀ Diagram

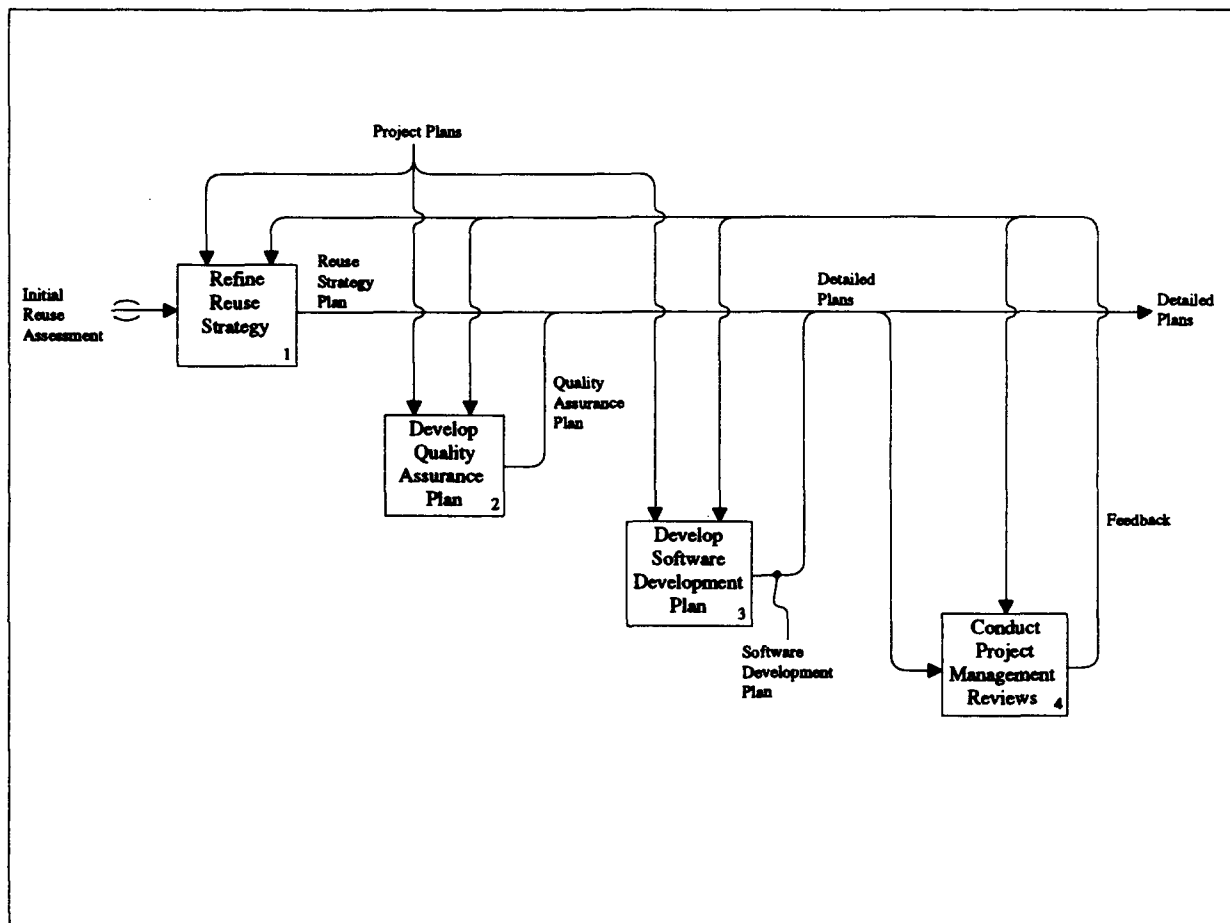
3.6.2.2.2 Develop Risk Mitigation Plan

Develop Risk Mitigation Plan processes reassess high risk areas and develop a plan to mitigate these risks. The plan may be based in part on the lessons learned from developing initial prototypes during requirements definition and analysis.

3.6.2.2.3 Develop Software Test Plan

Develop Software Test Plan processes develop the Software Test Plan (STP) which describes the formal qualification test plans for one or more CSCIs. The STP identifies the software test environment resources required for formal qualification testing (FQT) and provides schedules for FQT activities. In addition, the STP identifies the individual tests that are performed during FQT.

3.6.2.2.4 Engineer Basic Software Architecture

Figure 85: Refine Project Plans IDEF₀ Diagram

Engineer Basic Software Architecture processes develop the basic software architecture definition for the system. The Engineer Basic Software Architecture IDEF₀ diagram is shown in Figure 86. Engineer Basic Software Architecture includes the Engineer High-Level Architecture, Assess Technology Support, Establish Training Procedures, and Develop MMI Description processes.

Engineer High-Level Architecture

Engineer High-Level Architecture processes develop the high-level software architecture and develop architectural prototypes. The Engineer High-Level Architecture IDEF₀ diagram is shown in Figure 87. Engineer High-Level Architecture includes the Select and Develop High-Level Architecture, Develop SRS/IRS, Develop Architecture Prototypes, and Conduct SSR processes.

- **Select and Develop High Level Architecture** processes support high-level software architecture development through reuse, reverse engineering, and/or new devel-

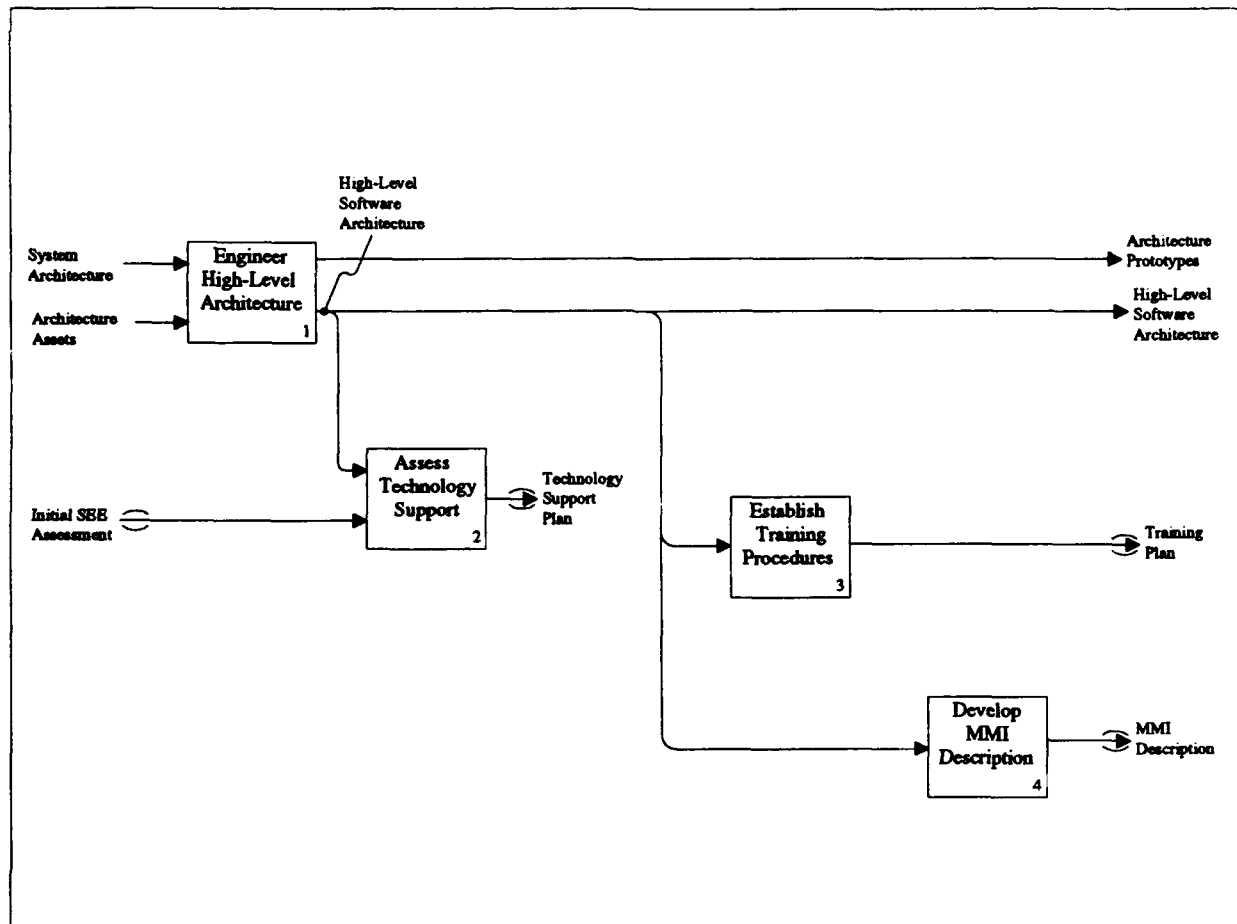


Figure 86: Engineer Basic Software Architecture IDEF₀ Diagram

opment. The Select and Develop High Level Architecture IDEF₀ diagram is shown in Figure 88. Select and Develop High Level Architecture includes the Select and Tailor High Level Architecture Assets, Reverse Engineer High Level Architecture, Develop Remaining High Level Architecture, and Integrate High Level Architecture processes

- **Develop SRS/IRS** processes support the development of the Software Requirements Specification (SRS) and the Interface Requirements Specification (IRS). The SRS specifies the engineering and qualification requirements for a CSCI and is used as the basis for the design and formal testing of the CSCI. The IRS specifies the requirements for one or more interfaces between one or more CSCIs and other configuration items or critical items. These work products are derived from the developed software architecture.
- **Develop Architecture Prototypes** processes develop architectural prototypes to assess and mitigate high risk areas.
- **Conduct SSR** processes support the Software Specification Review.

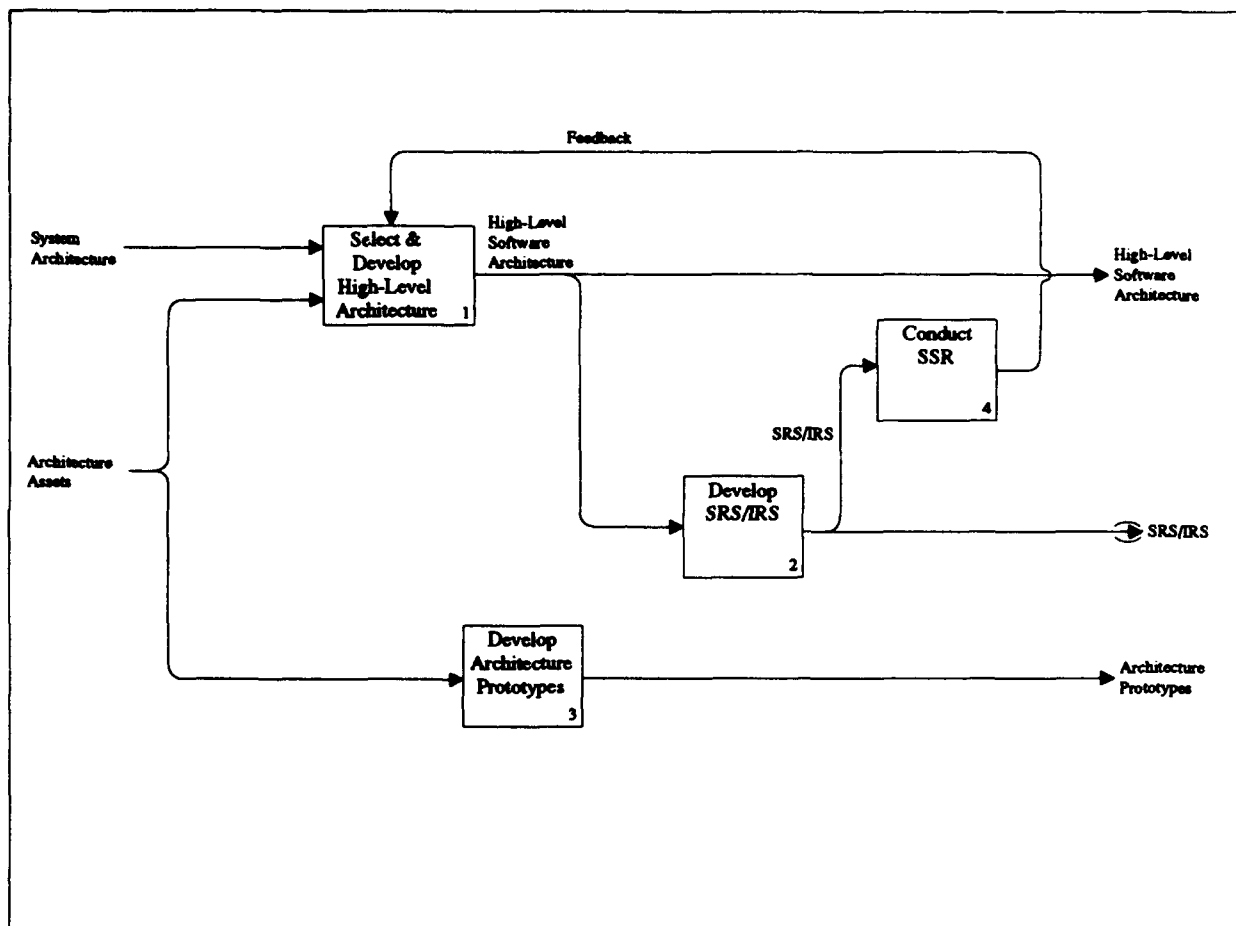


Figure 87: Engineer High-Level Architecture IDEF₀ Diagram

Assess Technology Support

Assess Technology Support processes assess the technology that can be used to support the system development, including software engineering environment support. The software engineering environment is also tailored, if necessary, to the project needs. These assessments are performed in the context of the software architecture, which imposes initial requirements on modeling tools and may also have some implications on the detailed design, implementation, and testing technology required.

Establish Training Procedures

Establish Training Procedures processes develop training procedures to train system users.

Develop MMI Description

Develop MMI Description processes describe the Man Machine Interface of the system. This

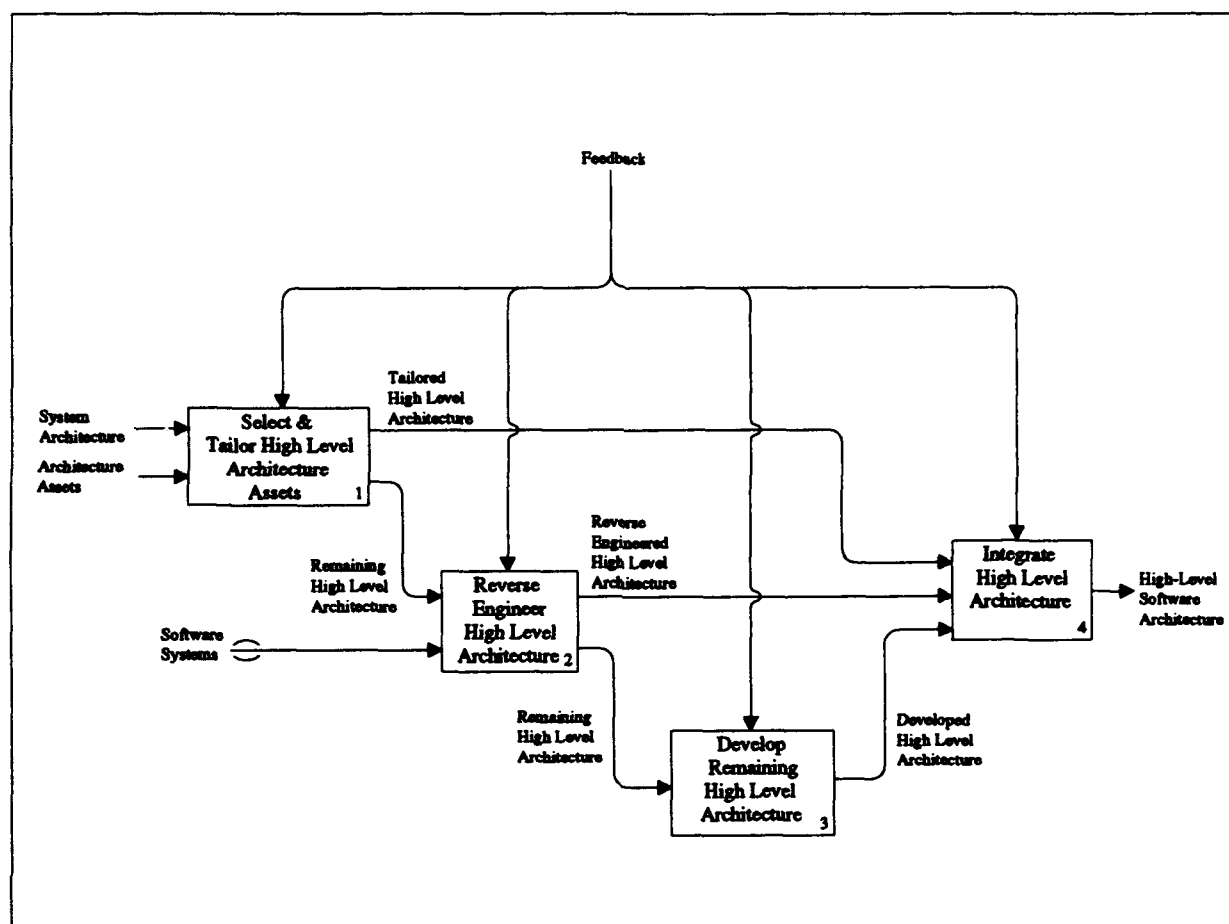


Figure 88: Select and Develop High Level Architecture Architecture IDEF₀ Diagram

human interface design may involve meeting with potential users and possibly developing rapid prototypes to determine characteristics of the interface that meet their needs.

3.6.2.3 Refine Architecture and Engineer Critical Elements

Refine Architecture and Engineer Critical Elements processes refine the high-level software architecture, develop prototypes for critical elements, and perform preliminary software design. The process abstraction hierarchy diagram for Refine Architecture and Engineer Critical Elements is shown in Figure 89.

The Refine Architecture and Engineer Critical Elements IDEF₀ diagram is shown in Figure 90. Refine Architecture and Engineer Critical Elements includes the Refine Project Plans, Assess Risks, Engineer Software Architecture, Engineer Critical Element Prototypes, and Perform Preliminary Design processes.

3.6.2.3.1 Refine Project Plans

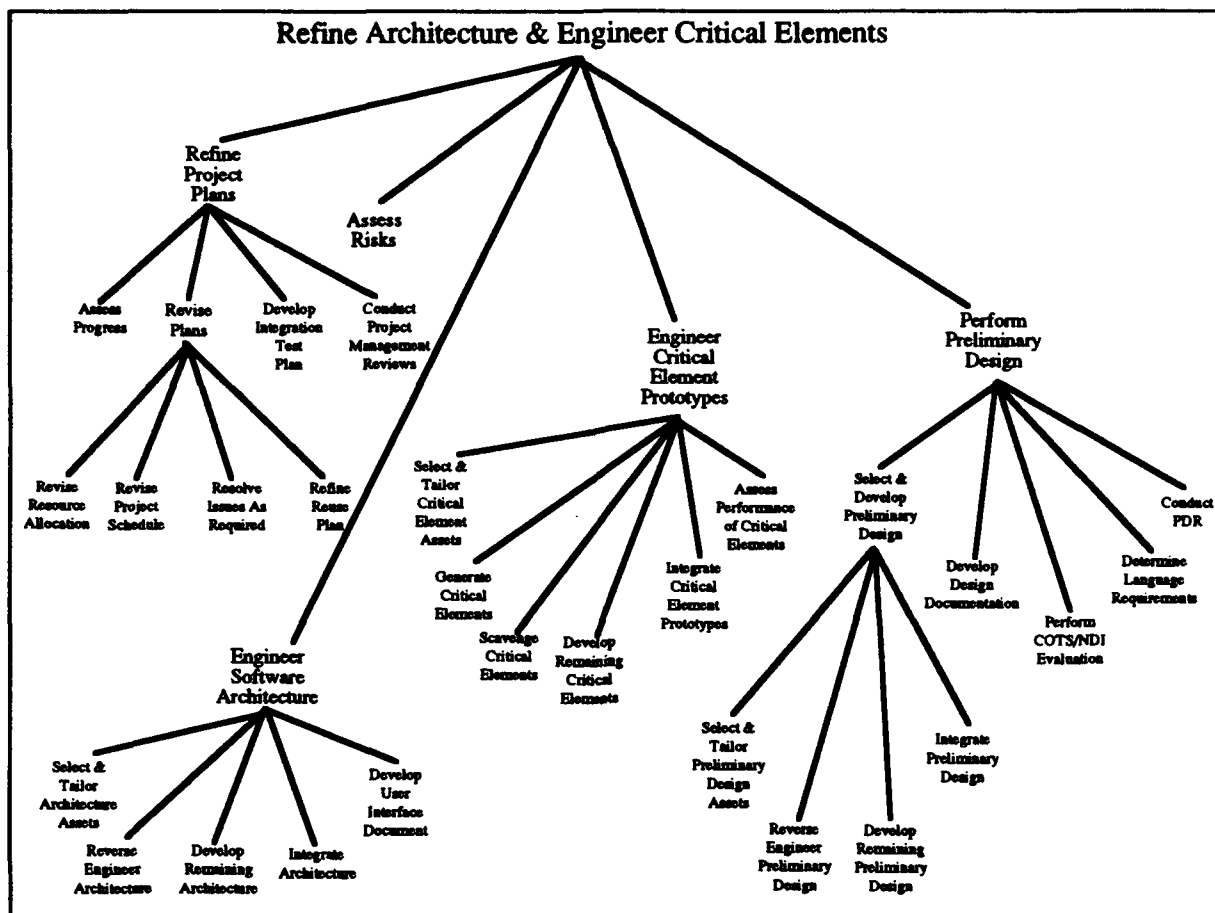


Figure 89: Refine Architecture and Engineer Critical Elements Process Abstraction Hierarchy

Refine Project Plans processes refine the project plans from the high-level architecture development phase. The Refine Project Plans IDEF₀ diagram is shown in Figure 91. Refine Project Plans includes the Assess Progress, Revise Plans, Develop Integration Test Plan, and Conduct Project Management Reviews processes.

Assess Progress

Assess Progress processes assess progress of the project to date against schedule and goals and make corrections if necessary.

Revise Plans

Revise Plans processes revise the plans developed in the requirements definition and high-level architecture development phases. The Revise Plans IDEF₀ diagram is shown in Figure 92. Revise Plans includes the Revise Resource Allocation, Revise Project Schedule,

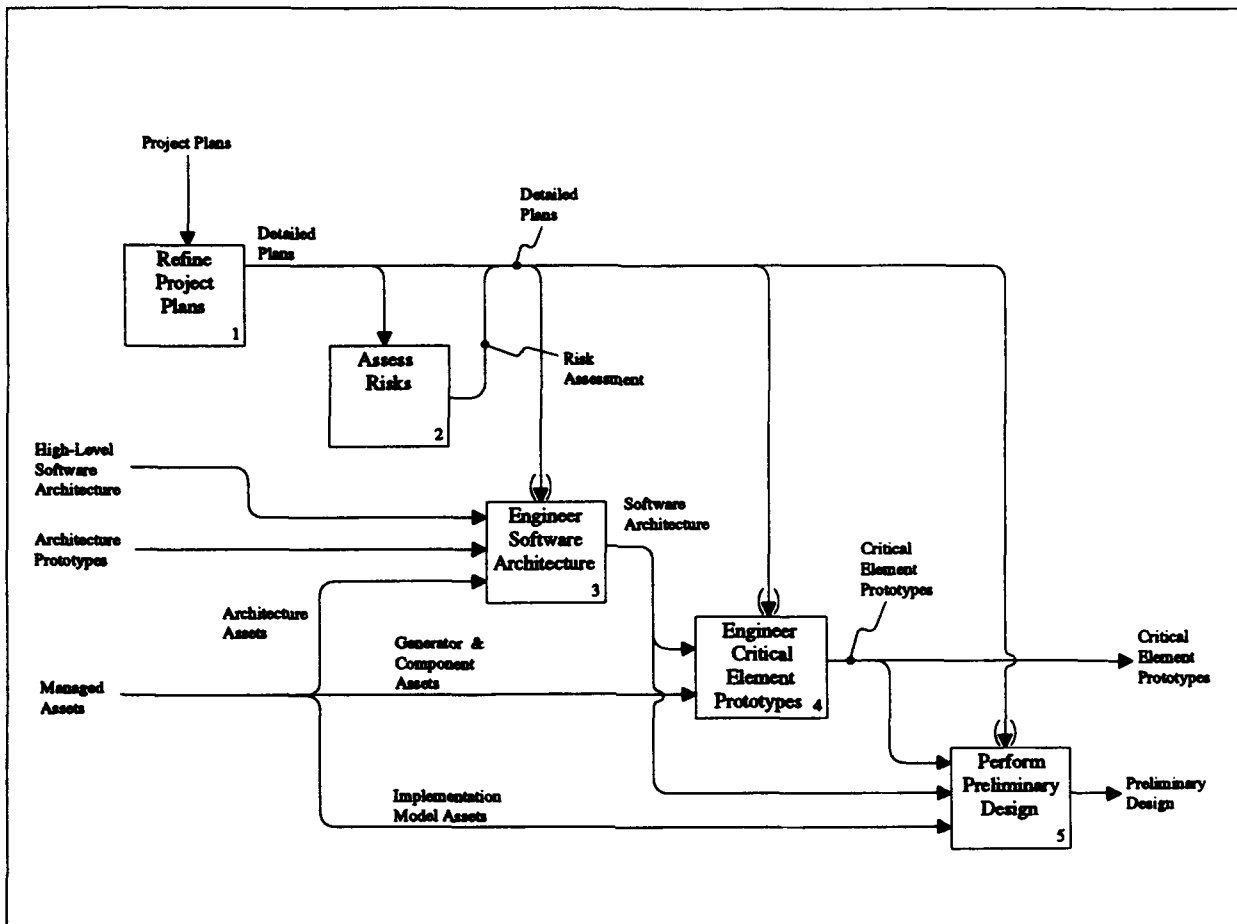


Figure 90: Refine Architecture and Engineer Critical Elements IDEF₀ Diagram

Resolve Issues As Required, and Refine Reuse Plan processes.

Develop Integration Test Plan

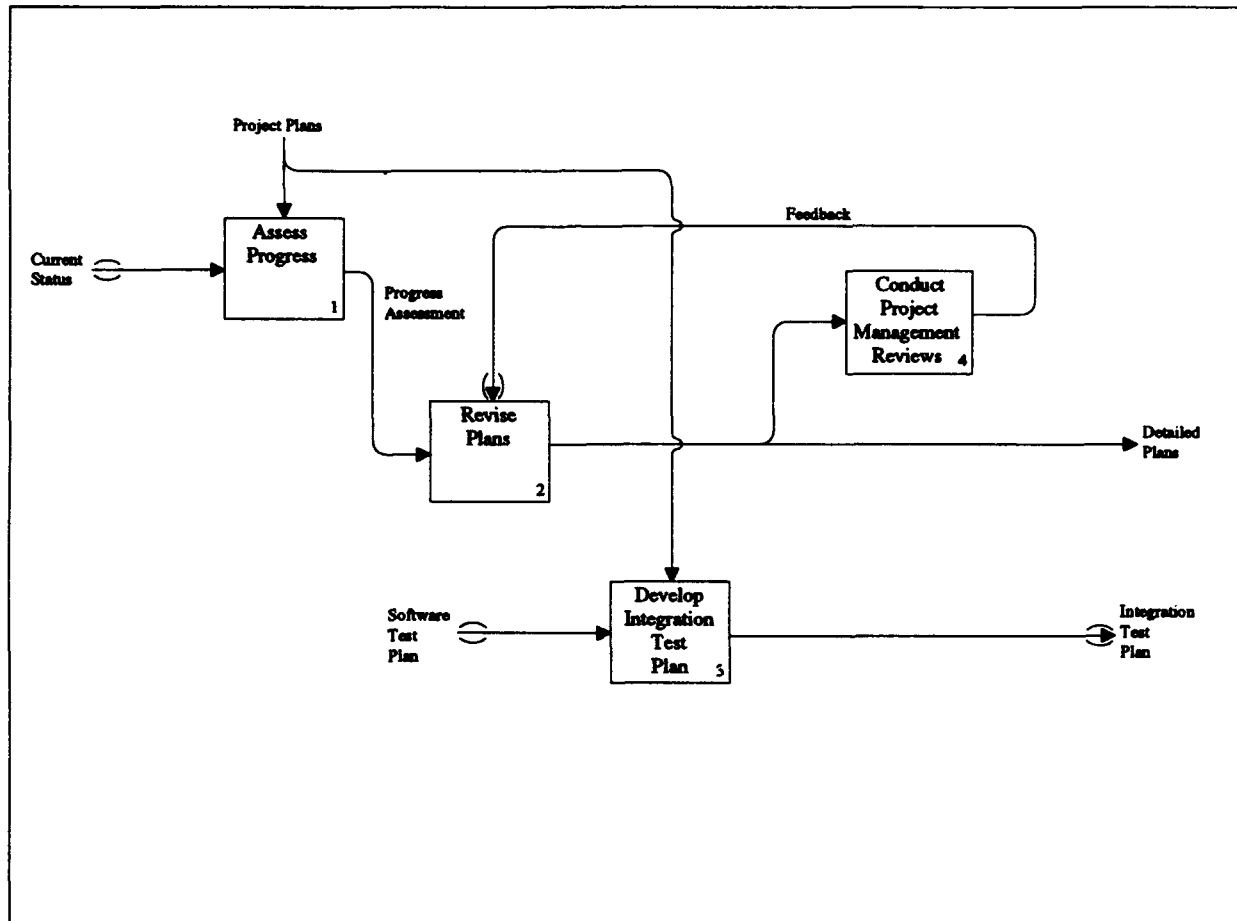
Develop Integration Test Plan processes develop the test plan for system integration and test.

Conduct Project Management Reviews

Conduct Project Management Reviews processes review project management activities.

3.6.2.3.2 Assess Risks

Assess Risks processes revise the risk assessments and risk mitigation plan created in previous development phases.

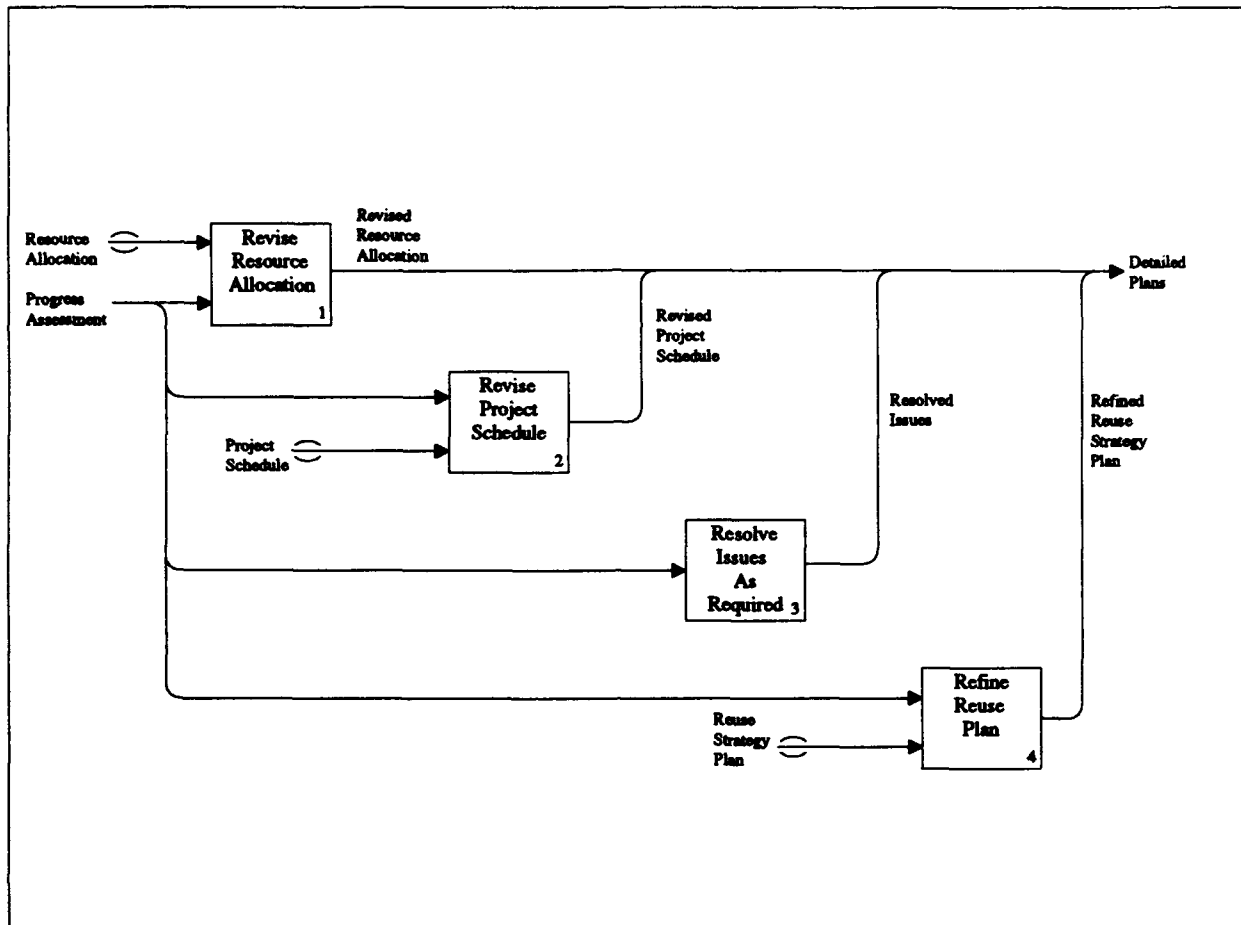
Figure 91: Refine Project Plans IDEF₀ Diagram

3.6.2.3.3 Engineer Software Architecture

Engineer Software Architecture processes refine the high-level software architecture. The Engineer Software Architecture IDEF₀ diagram is shown in Figure 93. Engineer Software Architecture includes the Select and Tailor Architecture Assets, Reverse Engineer Architecture, Develop Remaining Architecture, Integrate Architecture, and Develop User Interface Document processes. The first four activities involve the refinement of the architecture through reuse, reverse engineering, and/or new development. The User Interface Document defines the user interface of the system.

3.6.2.3.4 Engineer Critical Element Prototypes

Engineer Critical Element Prototypes processes develop prototypes for areas identified as critical software risks. By developing prototypes of these areas early, the risks can be lessened. The Engineer Critical Element Prototypes IDEF₀ diagram is shown in Figure 94. Engineer Critical Element Prototypes includes the Select and Tailor Critical Element Assets, Generate Critical Elements, Scavenge Critical Elements, Develop Remaining Critical

Figure 92: Revise Plans IDEF₀ Diagram

Elements, Integrate Critical Elements, and Assess Performance of Critical Elements processes.

The first five activities constitute the familiar “reuse, reverse engineering, new development” pattern of activity, with the possibility of generating the critical elements using an application generator being added. Assessing critical element performance is usually an important aspect of developing critical element prototypes. System performance is often one of the highest risk areas in software system development.

3.6.2.3.5 Perform Preliminary Design

Perform Preliminary Design processes develop the preliminary software design based on the software architecture. During preliminary design, reviews and walkthroughs are conducted as needed and all engineering notes should be documented. The Perform Preliminary Design IDEF₀ diagram is shown in Figure 95. Perform Preliminary Design includes the Select and Develop Preliminary Design, Develop Design Documentation, Perform COTS/NDI Evaluation, Determine Language Requirements, and Conduct PDR processes.

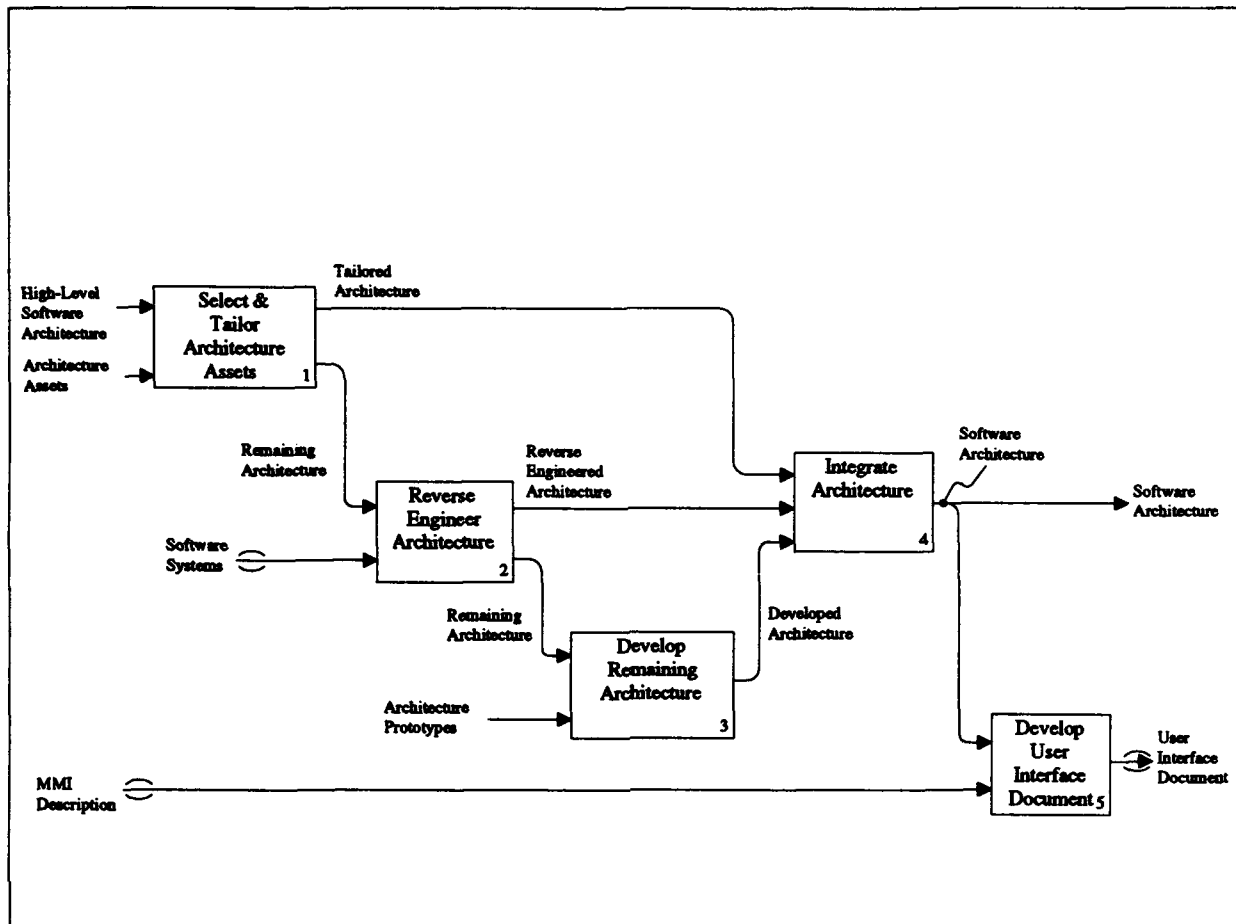


Figure 93: Engineer Software Architecture IDEF₀ Diagram

Select and Develop Preliminary Design

Select and Develop Preliminary Design processes develop the preliminary design through reuse, reverse engineering, and/or new development. The Select and Develop Preliminary Design IDEF₀ diagram is shown in Figure 96. Select and Develop Preliminary Design includes the Select and Tailor Preliminary Design Assets, Reverse Engineer Preliminary Design, Develop Remaining Preliminary Design, and Integrate Preliminary Design processes.

Develop Design Documentation

Develop Design Documentation processes develop the Software Design Document (SDD), Interface Design Document (IDD) and Software Development Folders (SDFs). The SDD describes the design of a CSCI, as composed of Computer Software Components (CSCs) and Computer Software Units (CSUs). The Interface Design Document (IDD) specifies the design for one or more interfaces between one or more CSCIs and other configuration items or critical items.

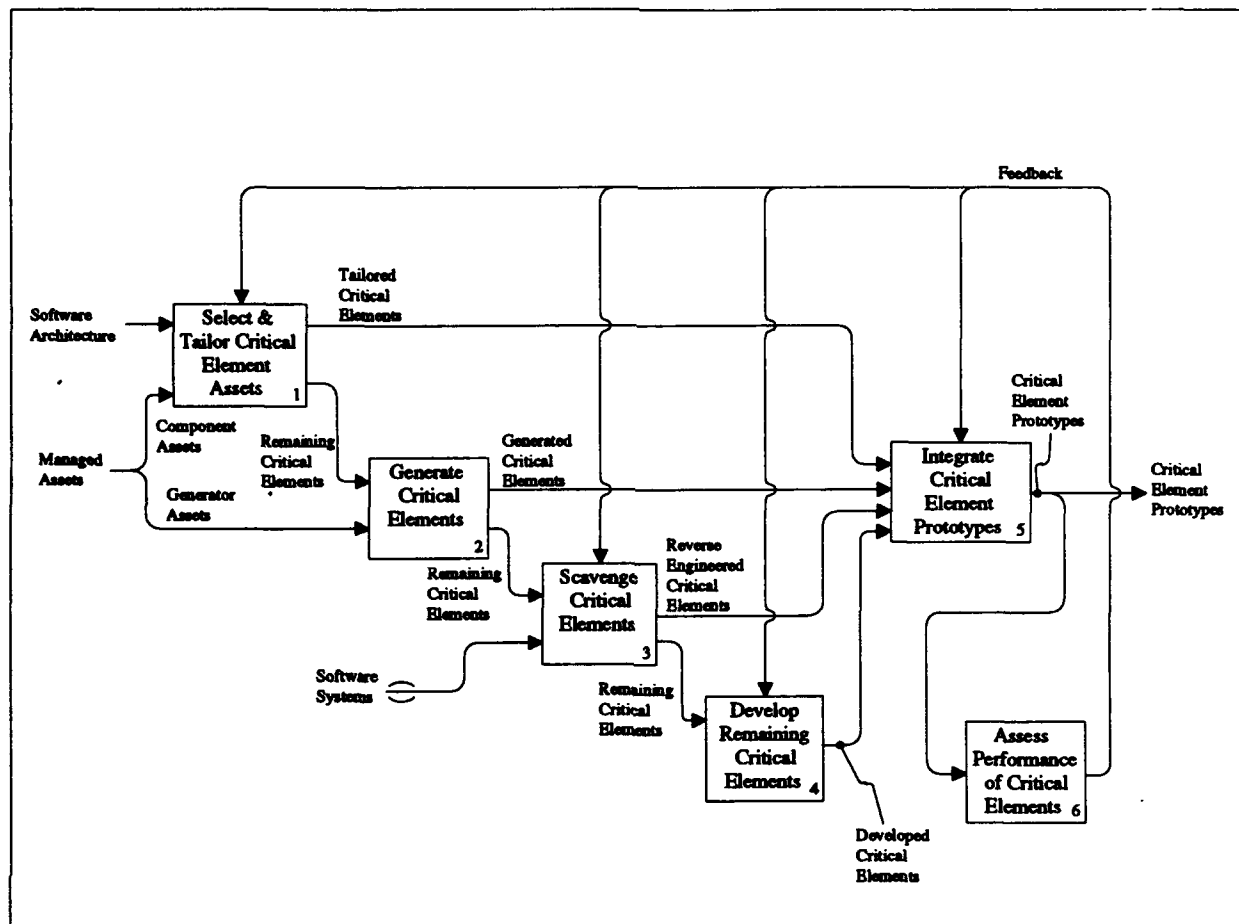


Figure 94: Engineer Critical Element Prototypes IDEF₀ Diagram

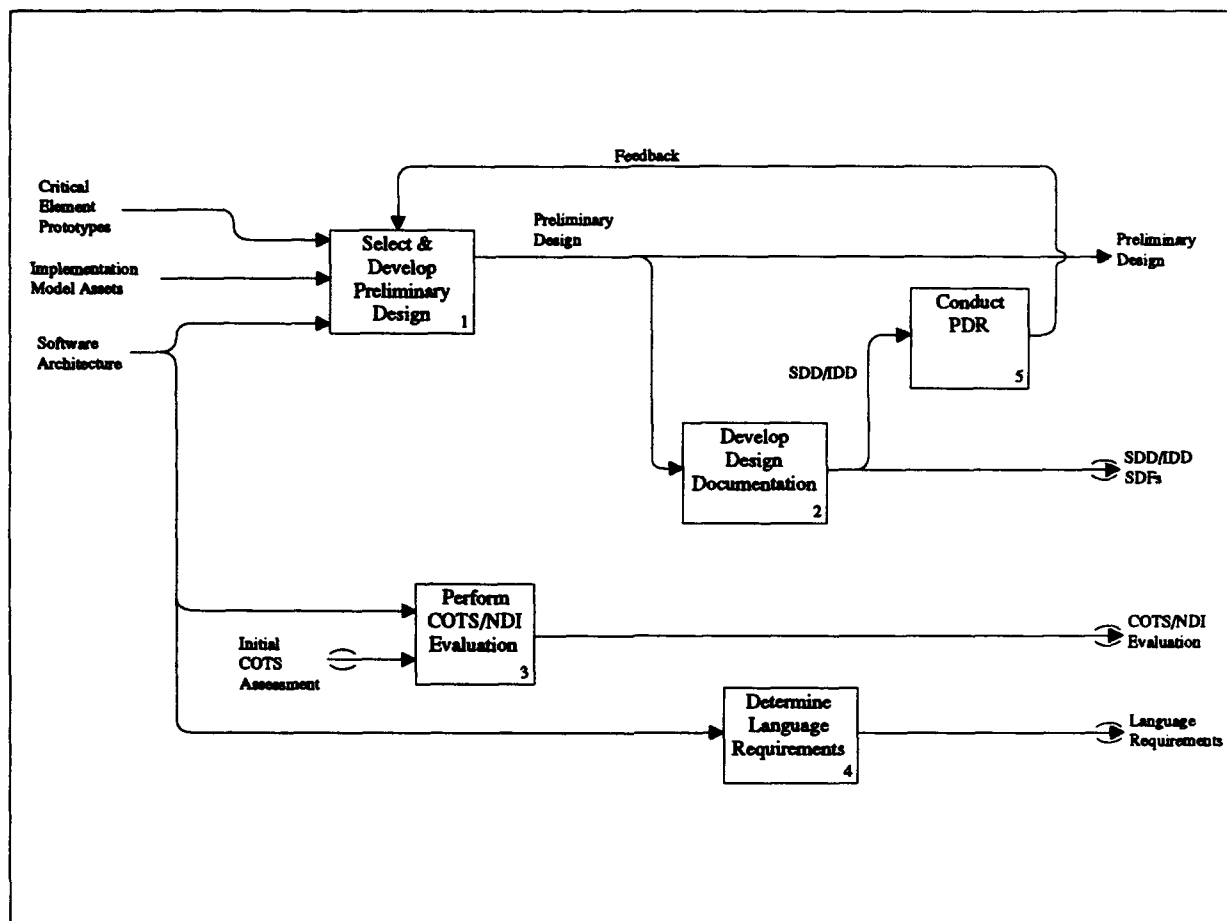
Perform COTS/NDI Evaluation

Perform COTS/NDI Evaluation processes evaluate where Commercial-Off-The-Shelf (COTS) and Non-Developmental Item (NDI) software can be effectively integrated into the system to fulfill some of the requirements. Ideally, this is mainly applicable where the software architecture or design diverges from or extends the architecture or design assets in the asset base. In areas where there is not divergence, Domain Engineering or Asset Management should have already identified the COTS/NDI items as relevant assets and they (or references to them) should already be in the asset library, ready to reuse.

Determine Language Requirements

Determine Language Requirements processes identify the requirements for the programming languages to be used during system implementation, and select appropriate languages.

Conduct PDR

Figure 95: Perform Preliminary Design IDEF₀ Diagram

Conduct PDR processes support the Preliminary Design Review.

3.6.2.4 Engineer System

Engineer System processes support the implementation, test, and integration of the software components of the system. During system engineering, reviews and walkthroughs are conducted as needed and all engineering notes should be documented. The process abstraction hierarchy diagram for Engineer System is shown in Figure 97.

The Engineer System IDEF₀ diagram is shown in Figure 98. Engineer System includes the Refine Project Plans, Assess Risks, Perform Detailed Design, Engineer Software Components, and Perform Testing and Integration processes.

3.6.2.4.1 Refine Project Plans

Refine Project Plans processes refine the project plans from the architecture refinement and

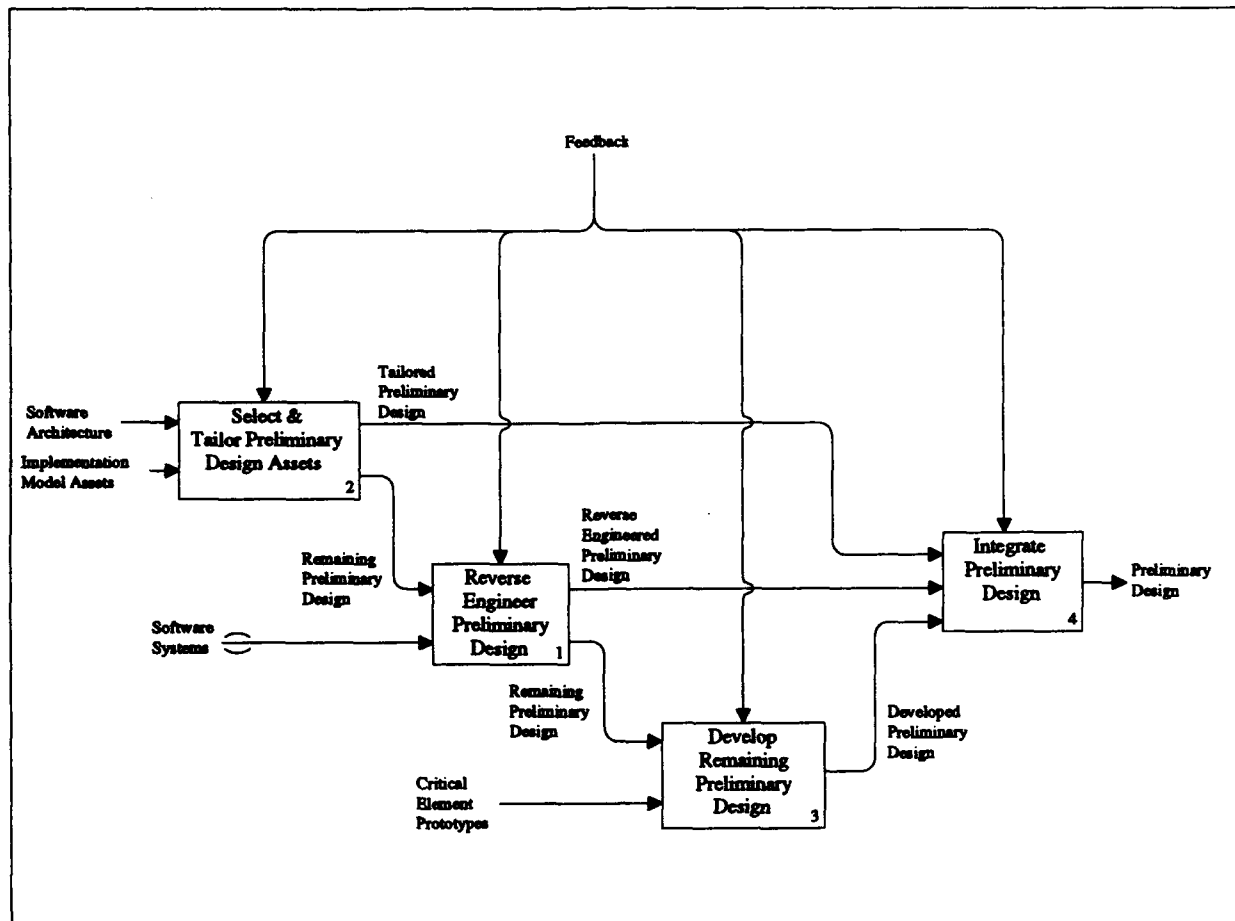


Figure 96: Select and Develop Preliminary Design IDEF₀ Diagram

critical element definition phase. The Refine Project Plans IDEF₀ diagram is shown in Figure 99. Refine Project Plans includes the Assess Progress, Revise Resource Allocation, Revise Project Schedule, Resolve Issues As Required, Develop Programming Standards, and Conduct Program Management Review processes.

3.6.2.4.2 Assess Risks

Assess Risks processes revise the risk assessments and risk mitigation plan defined previously, based on the results of critical element prototypes.

3.6.2.4.3 Perform Detailed Design

Perform Detailed Design processes develop the detailed software design. The Perform Detailed Design IDEF₀ diagram is shown in Figure 100. Perform Detailed Design includes the Select and Develop Detailed Design, Conduct Detailed Design Reviews and Walkthroughs, Develop Implementation Prototypes, Develop Event-Driven Prototypes, Develop Component

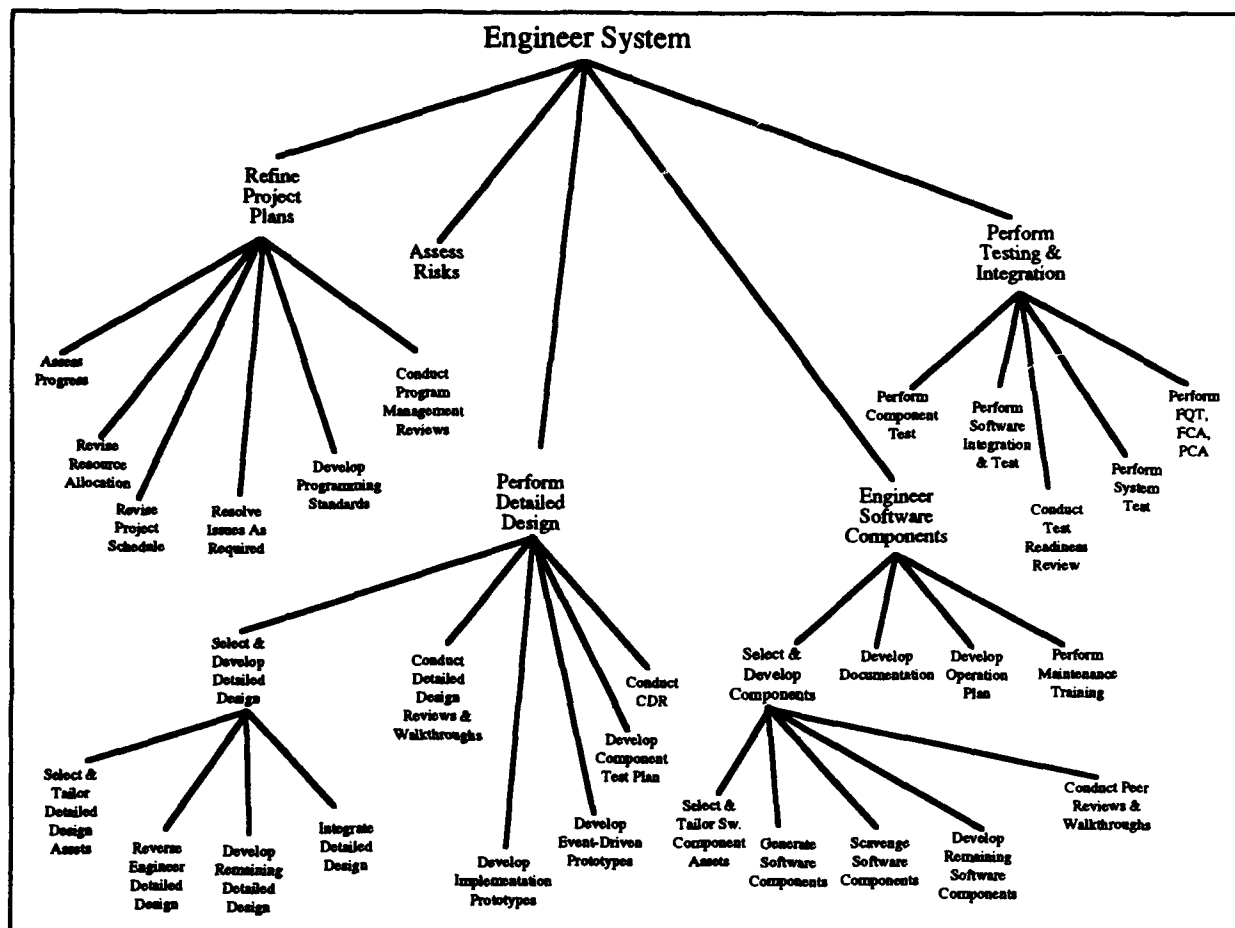


Figure 97: Engineer System Process Abstraction Hierarchy

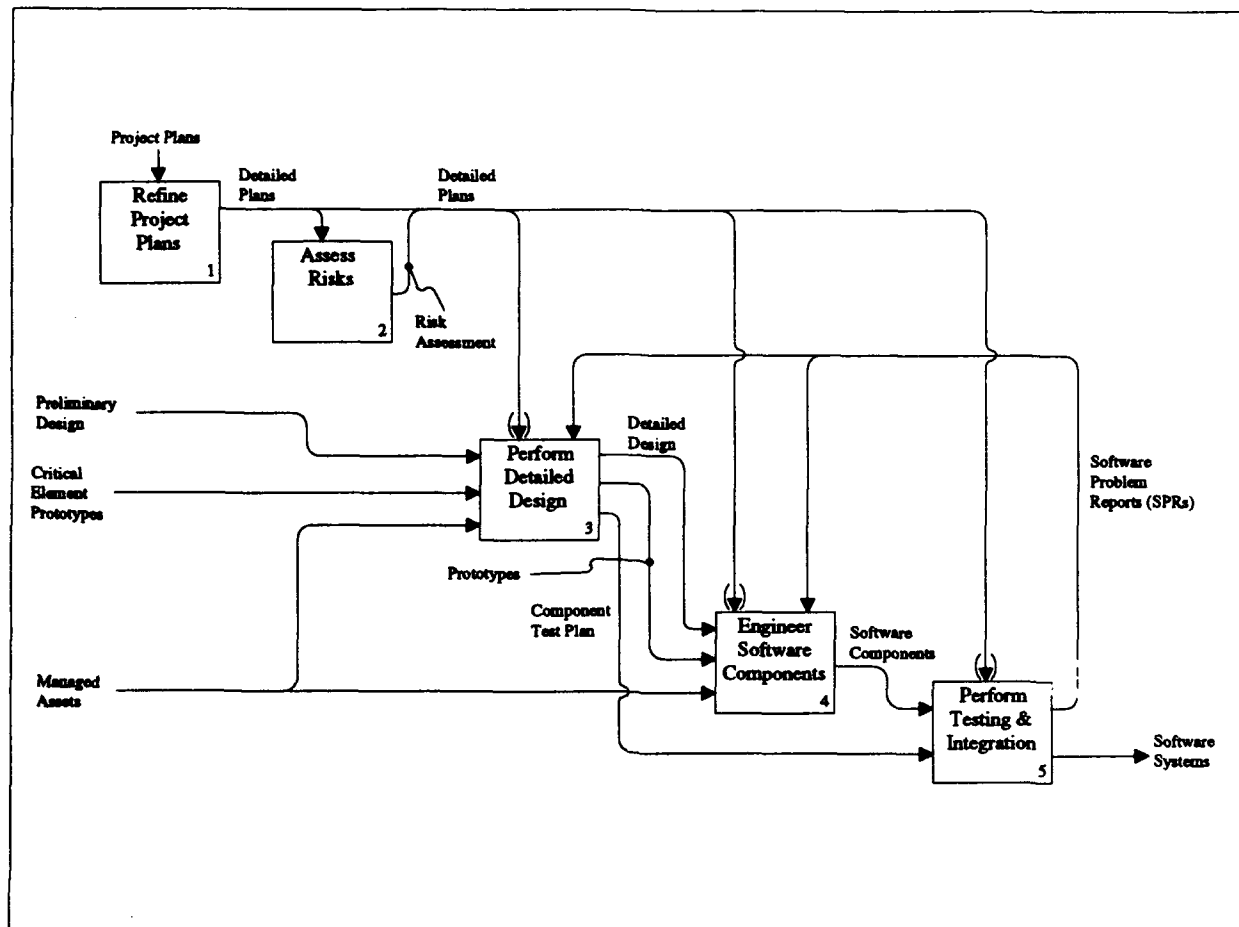
Test Plan, and Conduct CDR processes.

Select and Develop Detailed Design

Select and Develop Detailed Design processes develop the detailed design from the preliminary design, through reuse, reverse engineering, and/or new development. The Select and Develop Detailed Design IDEF₀ diagram is shown in Figure 101. Select and Develop Detailed Design includes the Select and Tailor Detailed Design Assets, Reverse Engineer Detailed Design, Develop Remaining Detailed Design, and Integrate Detailed Design processes.

Conduct Detailed Design Reviews and Walkthroughs

Conduct Detailed Design Reviews and Walkthroughs processes hold design reviews and walkthroughs to promote consensus among the entire engineering team on the software design.

Figure 98: Engineer System IDEF₀ Diagram**Develop Implementation Prototypes**

Develop Implementation Prototypes processes develop prototypes of software components based on high-risk areas.

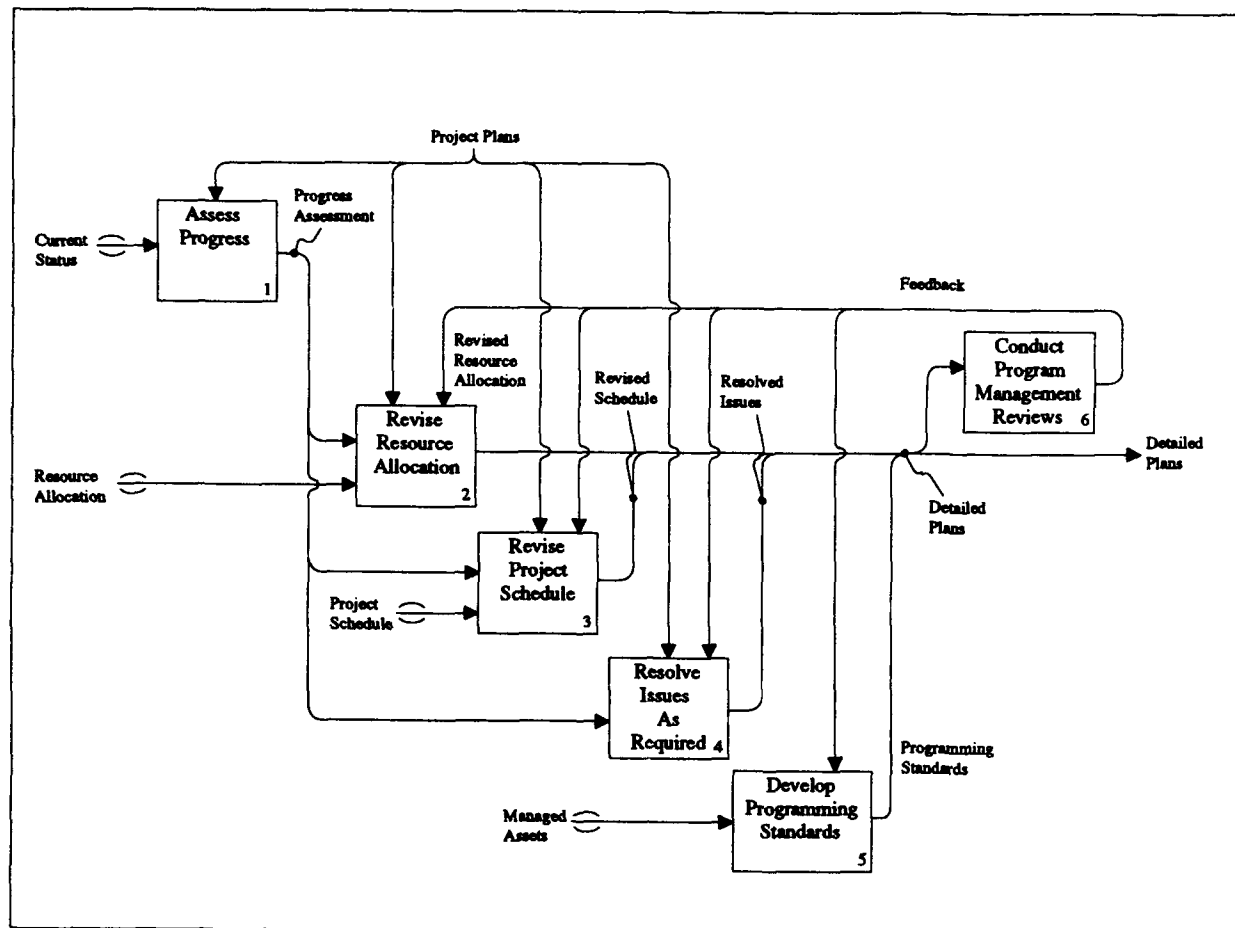
Develop Event-Driven Prototypes

Develop Event-Driven Prototypes processes develop event-driven prototypes in high-risk areas associated with concurrency and distributed processing.

Develop Component Test Plan

Develop Component Test Plan processes develop test plans for each software component.

Conduct CDR

Figure 99: Refine Project Plans IDEF₀ Diagram

Conduct CDR processes support the Critical Design Review.

3.6.2.4.4 Engineer Software Components

Engineer Software Components processes support the engineering of the software components from the detailed design. The Engineer Software Components IDEF₀ diagram is shown in Figure 102. Engineer Software Components includes the Select and Develop Components, Develop Documentation, Develop Operation Plan, and Perform Maintenance Training processes.

Select and Develop Components

Select and Develop Components processes reuse, generate, scavenge, or develop software components for the system. Peer review and walkthroughs are also conducted. The Select and Develop Components IDEF₀ diagram is shown in Figure 103. Select and Develop Components includes the Select and Tailor Component Assets, Generate Components, Scav-

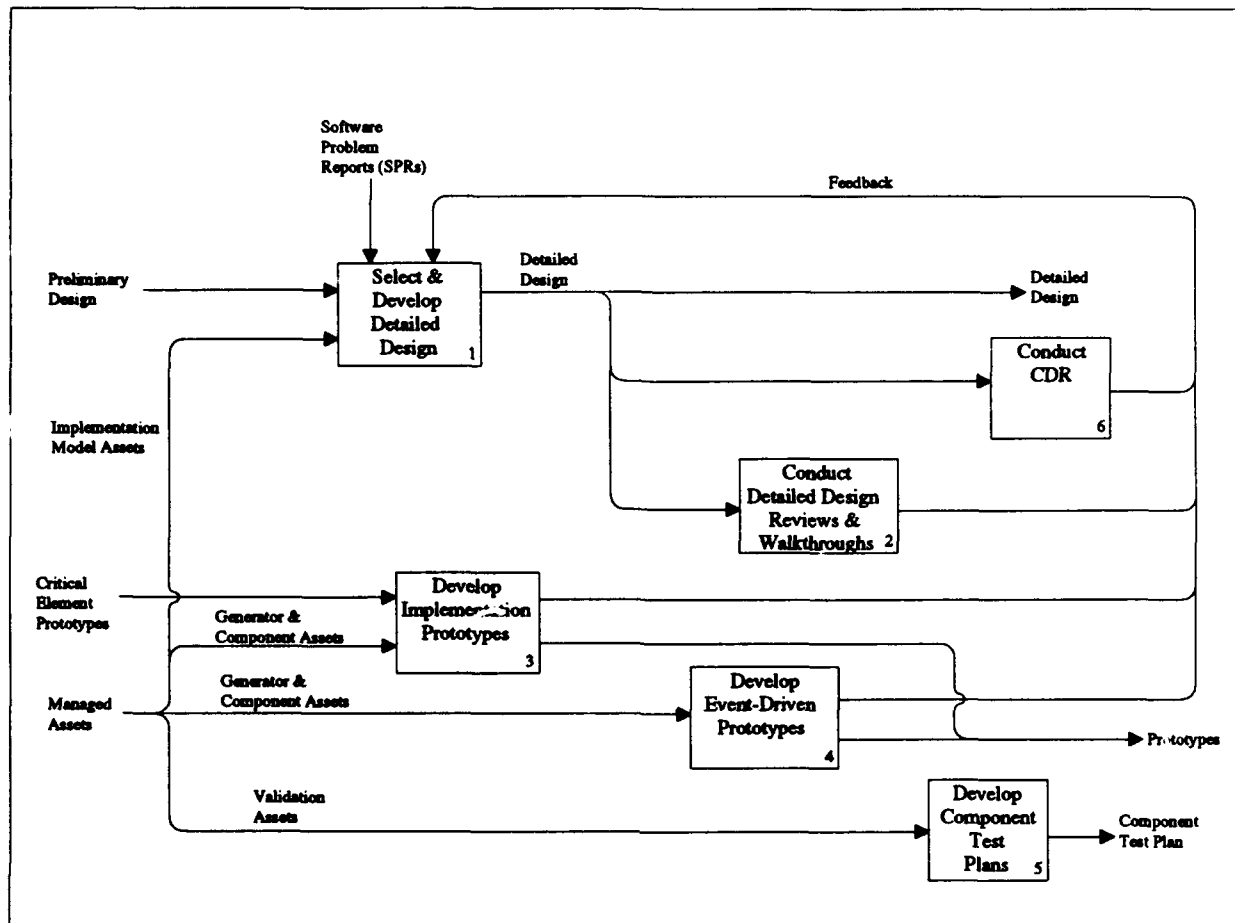


Figure 100: Perform Detailed Design IDEF₀ Diagram

Design Software Components, Develop Remaining Software Components, and Conduct Peer Reviews and Walkthroughs processes.

Develop Documentation

Develop Documentation processes develop user documentation, reusability documentation, and document engineering notes. Reusability documentation documents software components that appear to be good candidates for reuse in future systems. This information is passed to Asset Management so the candidate assets can be assessed for inclusion in the asset library. User documentation includes the Computer Resources Integrated Support Document (CRISD), Computer System Operator's Manual (CSOM), Software User's Manual (SUM), Software Programmer's Manual (SPM), Firmware Support Manual (FSM), Version Description Document(s) (VDDs), and Software Product Specification(s) (SPSs). The CRISD provides the information needed to plan for transitioning support of software to the maintainer. The CSOM describes how the computer system is operated. The SUM provides users with steps for executing the software, the expected output, and the mea-

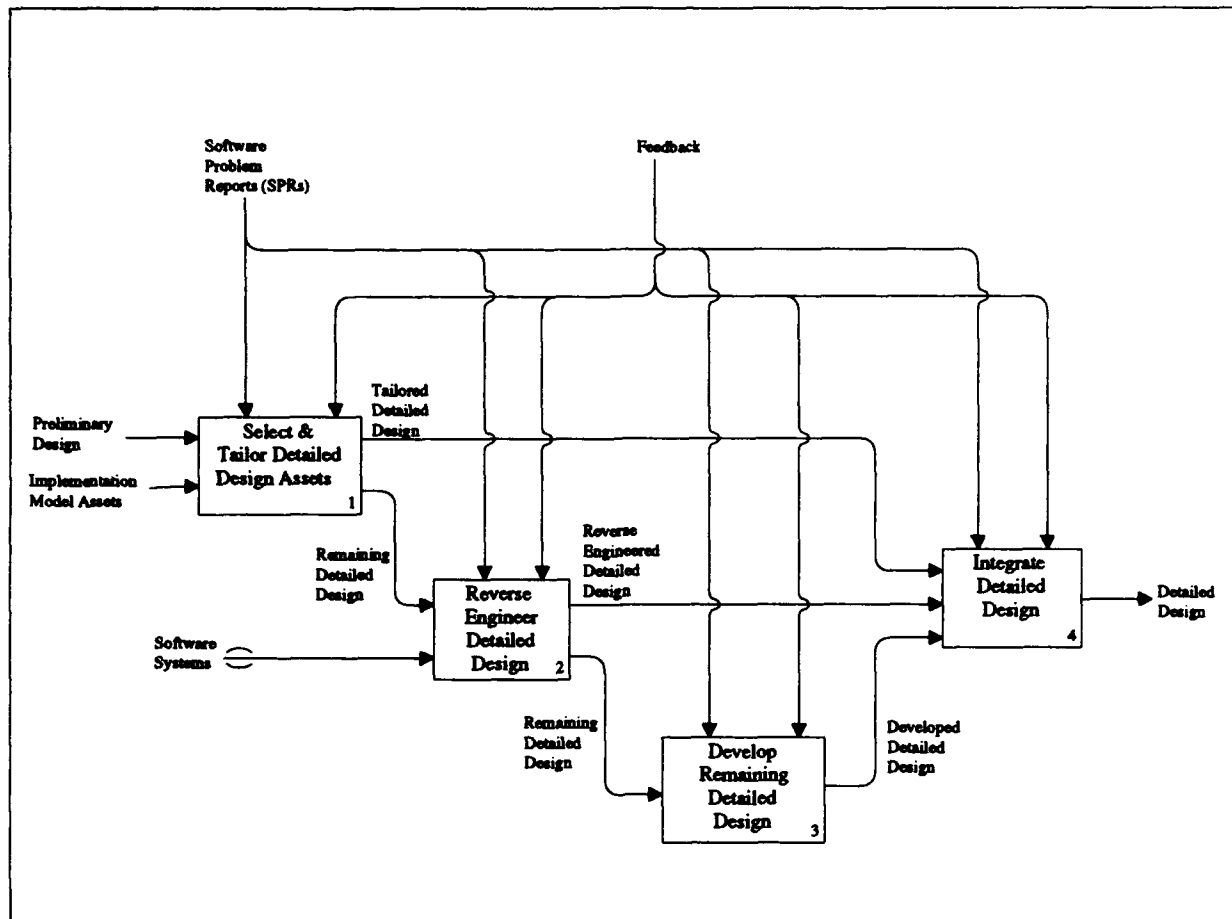


Figure 101: Select and Develop Detailed Design IDEF₀ Diagram

asures to be taken if error messages appear. The SPM provides information needed by a programmer to interpret, check out, troubleshoot, or modify the existing software, including documentation of the maintainability and evolvability of the system. The FSM provides the information necessary to load software or data into firmware components of the system. The VDD identifies and describes the version of the CSCI.

Develop Operation Plan

Develop Operation Plan processes develop the plan for installation and operation of the software at the user's site.

Perform Maintenance Training

Perform Maintenance Training processes train the maintainers of the system. Included in this training is any information pertinent to the maintainability and evolvability of the system. The training should include instruction on how to use the asset library, possibly

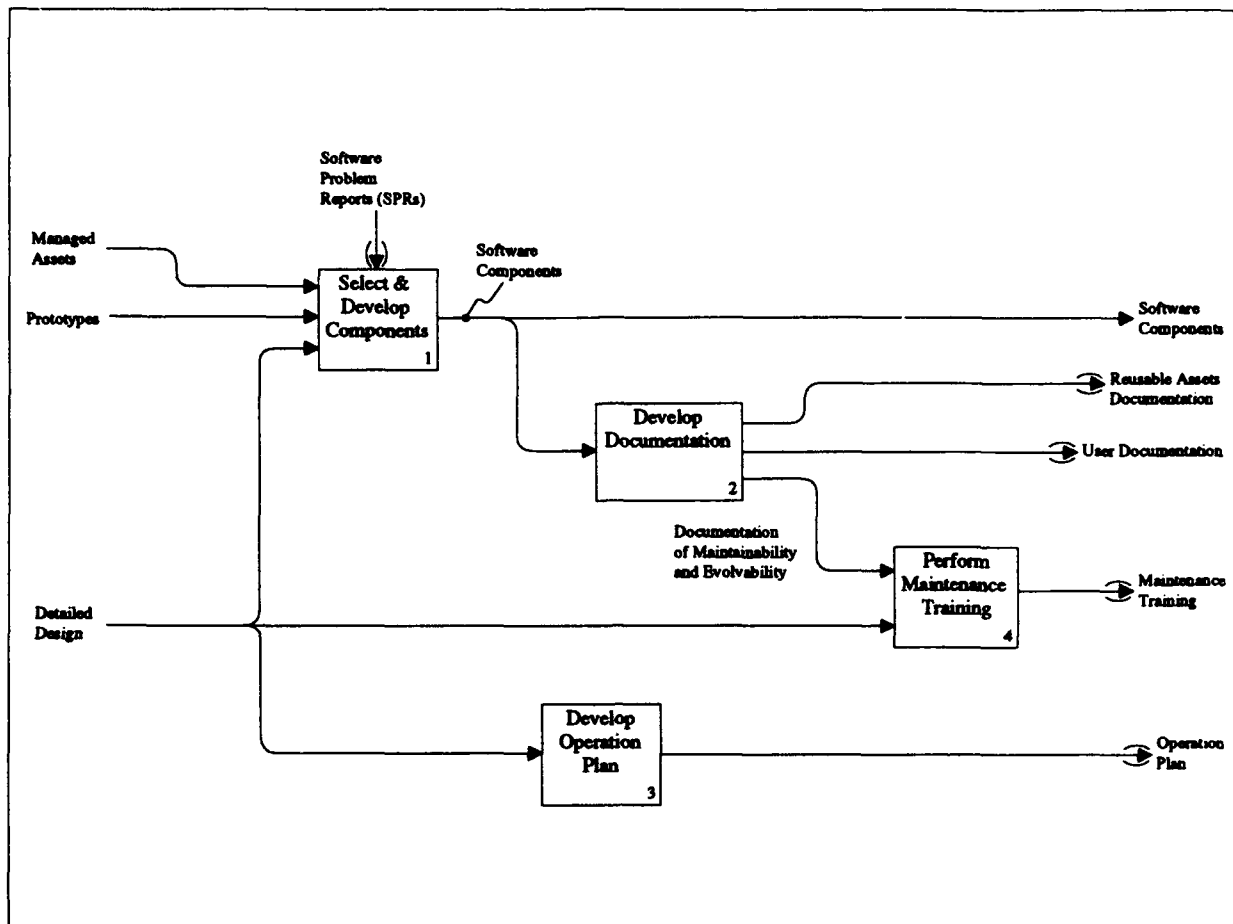


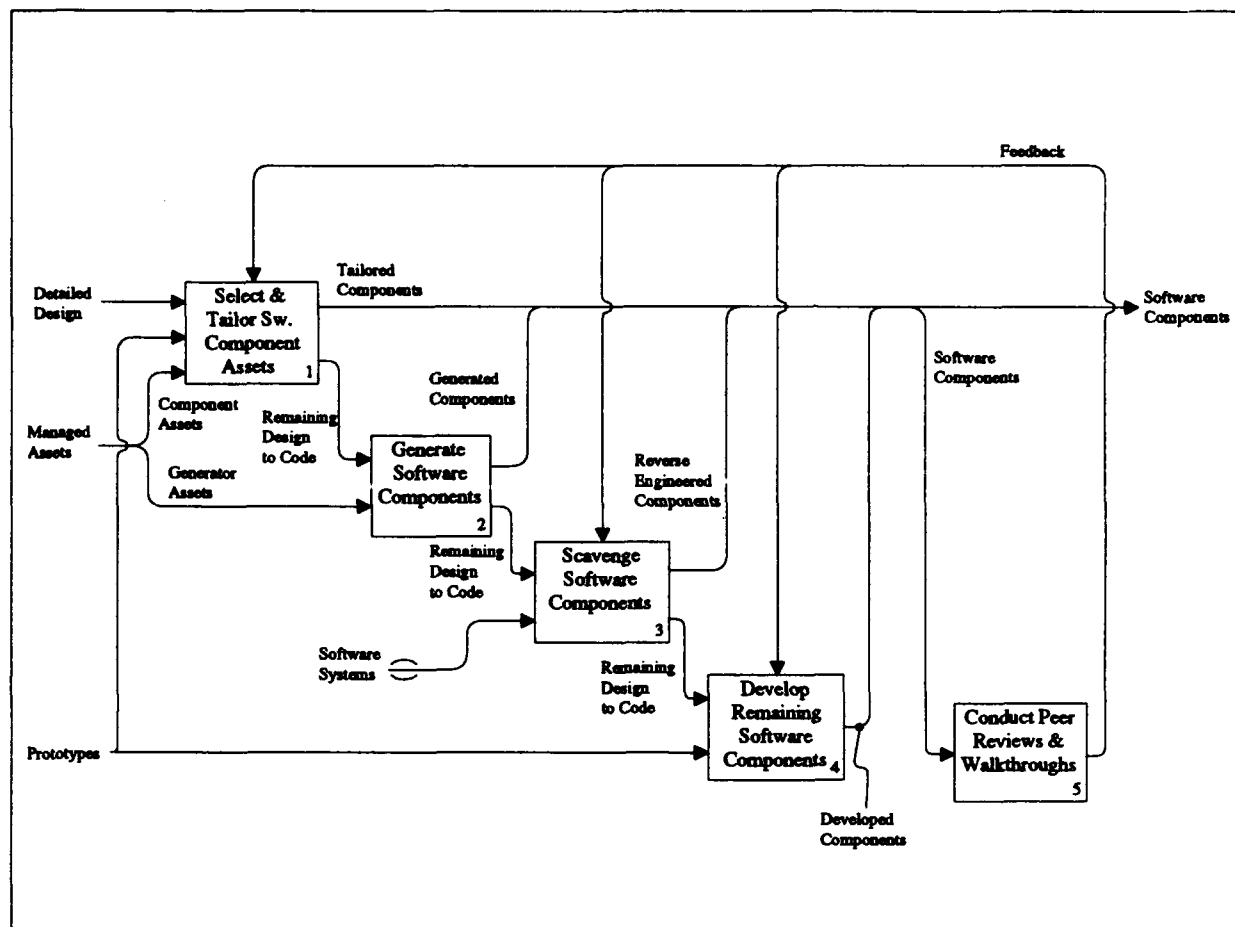
Figure 102: Engineer Software Components IDEF₀ Diagram

with high-level instruction on the domain model and the asset creation methods used.

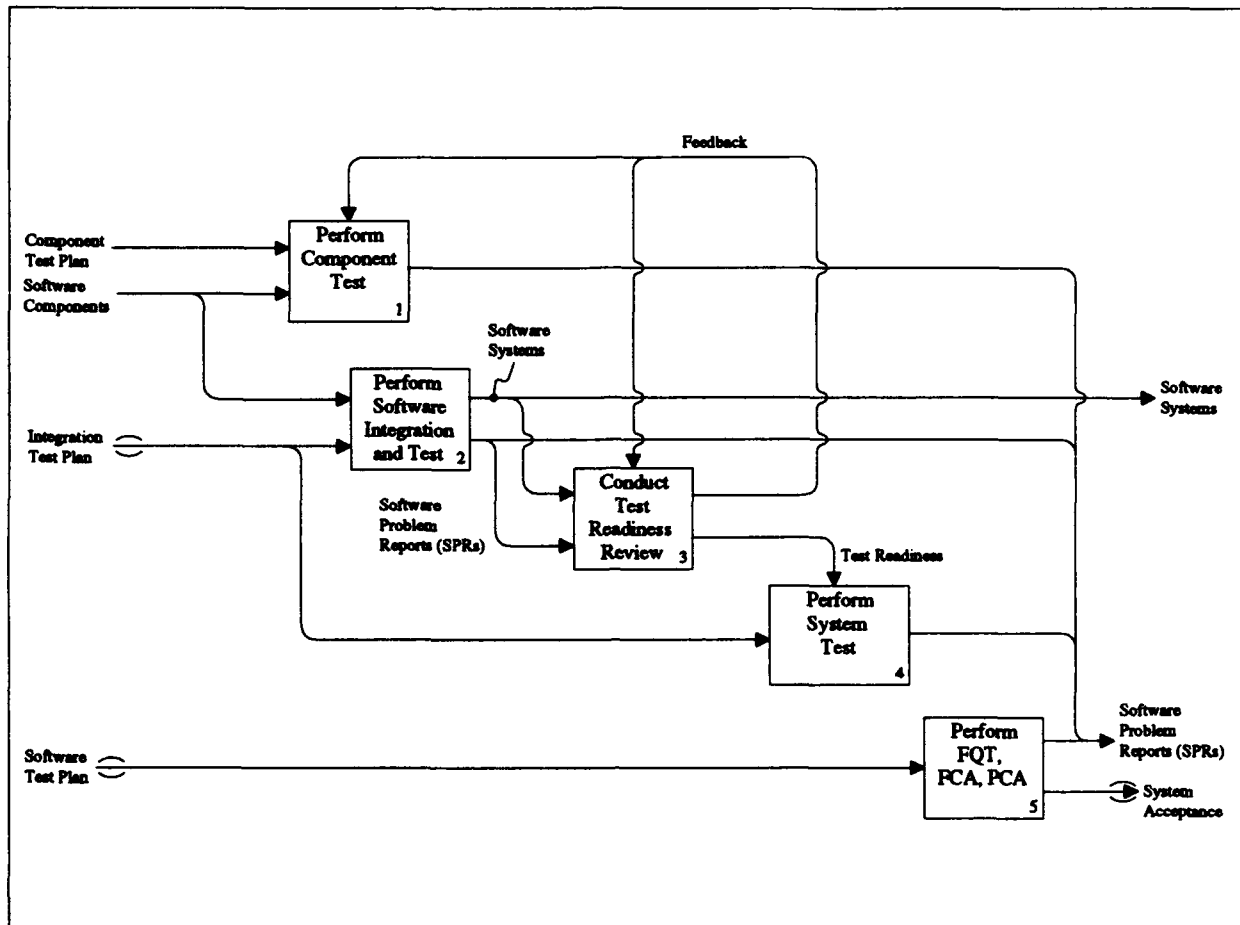
3.6.2.4.5 Perform Testing and Integration

Perform Testing and Integration processes integrate and test the software components. The Perform Testing and Integration IDEF₀ diagram is shown in Figure 104. Perform Testing and Integration includes the Perform Component Test, Perform Software Integration and Test, Conduct Test Readiness Review, Perform System Test, and Perform FQT, FCA, PCA processes.

- **Perform Component Test** processes test the individual software components. These tests are usually performed by the same software engineer who developed the component.
- **Perform Software Integration and Test** processes integrate software components into CSCs and CSCIs and test the conformance of the system to the requirements.

Figure 103: Select and Develop Components IDEF₀ Diagram

- **Conduct Test Readiness Review** processes determine whether the system is ready for final qualification testing.
- **Perform System Test** processes perform final testing of the entire system relative to the system requirements in preparation for the FQT, FCA, and PCA processes.
- **Perform FQT, FCA, PCA** processes support the performance of the Formal Qualification Testing (FQT), Functional Configuration Audit (FCA), and Physical Configuration Audit (PCA).

Figure 104: Perform Testing and Integration IDEF₀ Diagram

4 Applying the ROSE Process Model

The ROSE PM can be applied directly as a prescriptive model to carry out reuse-oriented software development and evolution. However, it is more likely that the ROSE PM will be adapted or tailored by an organization before being enacted. This section describes some of the ways an organization can apply the ROSE PM.

The ROSE PM can be applied to an organization in a manner similar to how the CFRP is applied. Since the ROSE PM is a more detailed specialization of the CFRP, it can be used more directly to establish a reuse-based process model in an organization. It can also be used to evaluate and compare other reuse-based process models in more detail than the CFRP. Since the ROSE PM extends the CFRP to include non-reuse-based portions of the software development and evolution life cycle, it does not have to be extended as the CFRP does in order to be applied in a general software development context. However, some tailoring will still be necessary based on the organization's particular circumstances.

4.1 Tailoring

The canonical ROSE PM presented in this document can be tailored for use as a process framework for reuse programs and projects within a line of business organization. In mapping the ROSE PM into an existing organization, reconfiguration, tailoring, addition, or deletion of processes within the model may be necessary.

Reconfiguring the ROSE PM

The standard canonic configuration of the ROSE PM presented in this document is but one possible configuration of the "ROSE concept". The ROSE concept focuses on several key principles, including:

- The CFRP Reuse Management (Plan-Enact-Learn) process idiom is a pervasive concept that can and should be applied at many different organizational levels.
- Software evolution and maintenance are inherent in the Reuse Management structure and do not have to be explicitly addressed as a separate phase in the life cycle.
- The CFRP Reuse Engineering (Create-Manage-Utilize) process families define distinct sets of reuse-based activities that can form the basis of individual projects within a reuse program.

The canonic ROSE PM can be reconfigured in a number of ways to produce alternative "ROSE-consistent" process models that remain generally faithful to the above principles,

The project submodels chosen in the canonic ROSE model are a logical grouping of activities. An organization can configure this logical grouping to more directly reflect the desired mode of operation or the structure of the organization. For example, an organization oriented towards R&D and innovation could "elevate" the Innovation Exploration process category to become an additional submodel that is used by individual projects. A submodel can also be "demoted"; for example, if Asset Management is viewed as being more of an infrastructure activity and less of a project-level activity, it can be integrated into the Infrastructure Implementation process category.

Adding Detailed Processes to the ROSE PM

The ROSE PM is termed a process "framework" because it does not include detailed descriptions of the processes used to carry out the activities. Detailed processes or methods can be identified, adapted, or defined by an organization and enacted within the ROSE framework to plan and perform reuse-based software engineering. Many ROSE PM processes represent traditional software engineering activities, and traditional methods can be "plugged in" to

those processes. For example, the Perform Detailed Design process in Application Engineering could use any arbitrary detailed design method. Other ROSE PM processes are more reuse-specific, and reuse-oriented methods must be applied.

In selecting or defining detailed methods, however, care must be taken to ensure that the methods used are compatible across process categories, families, and submodels. For example, either the methods used to develop domain assets should be directly compatible with the methods used in Application Engineering, or the Asset Management processes should ensure that the assets are filtered or presented in such a way that they are readily utilized by the application engineers.

Tailoring the ROSE PM to Existing Business Practices

The ROSE PM can also be tailored to an organization's existing structure or fundamental engineering practices. For example, ROSE activities can be performed across a number of levels of management or can be distributed across a number of peer organizations. The ROSE PM also assumes that Application Engineering follows DoD 2167A [15], but can be tailored to organizations following other standards. In general, this would be done by integrating Asset Utilization processes into another standard in a manner similar to how they are integrated into the existing ROSE PM Application Engineering submodel.

Tailoring the ROSE PM to Particular Management Styles

The ROSE PM is a hierarchically organized process model, with Plan, Enact, and Learn activities being carried out at many levels within the organization. At each level the Plan, Enact, and Learn processes will "inherit" planning decisions and artifacts from the higher level and may supplement them to address issues local to the lower level scope. The degree to which such information is inherited and supplemented or overridden can be established arbitrarily by any organization, making the ROSE PM highly adaptable to different management styles. For example, if a micro-management style is prevalent at the Organizational Management level, the individual project management activities will inherit many planning decisions from the higher level, leaving project managers little latitude in overall project planning. On the other hand, the project managers may have a looser management style and impose few constraints on detailed project activities beyond those inherited from the higher level, so that engineering staff members have significant flexibility in planning their own detailed work.

Using the ROSE PM to Transition to and Evolve Reuse Practices

The ROSE PM does not have to be applied to a mature reuse organization. The model is very flexible and can be used by an organization to transition to reuse-oriented software evolution at its own pace. Since the Organization Management and Application Engineering submodels of the ROSE PM encompass traditional management and software engineering activities, an

organization can begin adopting ROSE within a largely traditional management and software engineering context. The Domain Engineering and Asset Management submodels could be enacted by pilot reuse projects within the organization. As the organization becomes more reuse-oriented the transition can slowly be made to integrate more reuse into the organization's software engineering life cycle.

Also, any given ROSE instance will not remain static over time. As products mature, there may be some adjustments to the activities that are performed to evolve them, relative to the initial development activities. This will generally involve emphasis or de-emphasis of particular activities, within the same overall activity structure.

4.2 Other Uses of the ROSE PM

In addition to being applied directly as the basis of an organization's process model, the ROSE PM can be used as an example of a CFRP-derived process model. It can thus be used to guide the derivation of alternative process models from the CFRP that better meet the needs of the organization.

The ROSE PM can also be used as a basis for assessing the reuse practices already established or planned within an organization. By comparing these reuse practices to the activities described in the ROSE PM, areas of deficiency in these practices can be found.

In addition, the ROSE PM can be used as a basis for improving understanding of reuse-based software engineering. A reuse advocate can use the understanding gained from the ROSE PM to promote the adoption of reuse within an organization.

5 Outstanding Issues

This section identifies open issues with respect to the ROSE PM. These issues merit further consideration but could not be resolved at this time due to schedule constraints and limited opportunity for review and validation. These issues will be addressed in the future through further examination, review, and trial usage. The ROSE PM will evolve accordingly, and enhancements will be reflected in future versions of this document.

The following list of issues is representative of the issues that have been identified. It includes all the most important known issues, but should not be considered exhaustive.

- The ROSE PM should be consistent throughout in terms of completeness, and level of abstraction (or generality). The level of abstraction should be chosen carefully to ensure that the model describes processes in enough detail to be useful, while not unnecessarily limiting their adaptability. In this sense, the ROSE PM should be subjected to many of the same kinds of analyses that are undertaken in the design of any reusable asset. One example aspect of the model that should be examined from this perspective is the specificity of the approach described in the Application Engineering submodel. It may be preferable to generalize aspects of the process that seem overly prescriptive, such as the Critical Elements activities or the overall 2167A orientation of the process.
- The names of some processes in the ROSE PM, and their grouping and ordering in particular process categories or families, may be questionable or debatable. In general, this may not be a major problem, since the model is intended to be more suggestive than prescriptive, and organizations using the model can adapt it to address such concerns as needed. However, the model will be most useful if the processes are clearly described and organized and are consistent with practical usage. Some of the specific issues that have already been noted in this regard include:
 - The relationships of the various levels of planning and scoping should be clarified, and some of the specific activities should be renamed or reorganized. For example, the low-level project planning activities in Application Engineering could benefit from clarification, and the location of domain definition and scoping activities in Analyze and Model Domain merits reexamination.
 - It is preferable to define a Develop Implementation Model process category in Domain Engineering, with the Develop Application Generators and Develop Software Components processes embedded within it.
 - The Forecast Requirements process in Plan Domain Analysis should probably be merged with the Analyze Future Requirements process in Acquire Domain Knowledge.
 - The relationships between system and software architecture engineering in Application Engineering should be clarified, and Engineer System Architecture should perhaps be included as a separate process category, rather than embedded in the Perform Requirements Definition and Analysis category.

- Some process names that are currently the same as the names of parent or peer processes should be made unique.
- The recommended ROSE PM approach to address development, maintenance and reengineering from a domain engineering perspective needs to be further clarified and elaborated.
- The planning and learning activities in Domain Engineering, Asset Management, and Application Engineering should be refined to be more specific to each submodel.
- In general, the process categories include few learning activities, and additional learning activities should be included as appropriate.
- Similarly, additional metrics collection activities should be included in the process categories as appropriate.
- Other relevant process models should be reviewed in more detail and the ROSE PM should evolve to be consistent or compatible with those other models, as appropriate. Examples of models to consider include DoD MIL-STD-SDD [16] and the domain engineering process described in the Virginia Center of Excellence (VCOE) Domain Engineering Guidebook [9].
- Guidance in how to apply the ROSE PM is currently limited. More feedback is needed from trial users of the ROSE PM to further clarify issues associated with the practical application of the model. This will suggest enhancements to the model and enable development of more extensive guidance for applying the model.
- This document should be structured and presented more clearly to better communicate the structure and content of the ROSE PM. This may involve, among other things, simplifying or clarifying the terminology used. It should also provide a more complete description of the model by, for example, describing the Develop Domain Model process in more detail.

References

- [1] Victor Basili. "Viewing Maintenance as Reuse-Oriented Software Development." *IEEE Software*, 7(1):19-25, January, 1990.
- [2] Victor Basili. "Special Tutorial Session: Starting an Experience Factory." *Seventeenth Annual Software Engineering Workshop*, December 1992.
- [3] Victor Basili, Gianluigi Caldiera, and Giovanni Cantone. "A Reference Architecture for the Component Factory." *ACM Transactions on Software Engineering and Methodology*, 1(1):53-80, January, 1992.
- [4] Ted Biggerstaff. "Design Recovery for Maintenance and Reuse." *IEEE Computer*, 22(7):36-49, July, 1989.
- [5] K.C. Kang, S.G. Cohen, J.A. Hess, W.E. Novak, and A.S. Peterson. *Feature-Oriented Domain Analysis (FODA) Feasibility Study*. Technical Report CMU/SEI-90-TR-21, Software Engineering Institute, Carnegie Mellon University, Pittsburgh, PA, November 1990.
- [6] David A. Marca and Clement L. McGowan. *SADT : Structured Analysis and Design Techniques*. McGraw-Hill, New York, NY, 1988.
- [7] Software Technology for Adaptable Reliable Systems (STARS). "Reuse Library Process Model." IBM STARS Technical Report CDRL 03041-001, US Air Force Materiel Command, Electronic Systems Center, Hanscom Air Force Base, MA, July 1991.
- [8] Softech, Inc. *Integrated Computer-Aided Manufacturing (ICAM) Architecture Part II. Volume IV - Function Modeling Manual (IDEF₀)*. AFWAL-TR-81-4023 V. IV, Materials Laboratory (AFWAL/MLTC), AF Wright Aeronautical Laboratories (AFSC), Wright-Paterson AFB, Ohio, June, 1981; available through IDEF Users Group.
- [9] Virginia Center of Excellence for Software Reuse and Technology Transfer (VCOE). *Domain Engineering Guidebook*, Version 01.00.03. Technical Report SPC-92019-CMC, Software Productivity Consortium, Herndon, Virginia, December 1992.
- [10] Software Technology for Adaptable Reliable Systems (STARS). *Composite Paradigm Report*. Paramax STARS Technical Report STARS-SC-03068/001/00, US Air Force Systems Command, Electronic Systems Division, Hanscom Air Force Base, MA, November 1990.
- [11] Software Technology for Adaptable Reliable Systems (STARS). *Organization Domain Modeling (ODM), Volume I - Conceptual Foundations, Process and Workproduct Descriptions*, Version 0.5 - DRAFT. Paramax STARS Technical Report STARS-UC-05156/024/00, US Air Force Systems Command, Electronic Systems Division, Hanscom Air Force Base, MA, July 1993.

- [12] Software Technology for Adaptable Reliable Systems (STARS). *Reuse Concepts Volume I - Conceptual Framework for Reuse Processes (CFRP)*, Version 2.0. Paramax STARS STARS-UC-05159/001/00, US Air Force Systems Command, Electronic Systems Division, Hanscom Air Force Base, MA, November 1992.
- [13] Software Technology for Adaptable Reliable Systems (STARS). *STARS Conceptual Framework for Reuse Processes (CFRP): Application*, Version 0.5. Paramax STARS STARS-UC-05159/002/00, US Air Force Systems Command, Electronic Systems Division, Hanscom Air Force Base, MA, July 1993.
- [14] Software Technology for Adaptable Reliable Systems (STARS). *Risk-Reduction Reasoning-Based Development Paradigm Tailored to Navy C² Systems Final Report*, Paramax STARS Technical Report STARS-SC-03070/001/00, US Air Force Systems Command, Electronic Systems Division, Hanscom Air Force Base, MA, January 1991.
- [15] United States Department of Defense. *Military Standard: Defense System Software Development*. 29 February 1988. DOD-STD-2167A.
- [16] United States Department of Defense. *Military Standard: Software Development and Documentation*, Draft. 22 December 1992. MIL-STD-SDD(DRAFT).